INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Bahia

Campus
Vitória da Conquista

COEEL
Coordenação de
Engenharia Elétrica

COORDENAÇÃO DE ENGENHARIA ELÉTRICA - COEEL

PROJETO FINAL DE CURSO - PFC

Controle Adaptativo de Tráfego: Um Estudo Comparativo entre Aprendizado por Reforço e Semáforos de Tempo Fixo

JEFERSON CAIO OLIVEIRA SILVA

Vitória da Conquista - BA

13 de agosto de 2024

JEFERSON CAIO OLIVEIRA SILVA

Controle Adaptativo de Tráfego: Um Estudo Comparativo entre Aprendizado por Reforço e Semáforos de Tempo Fixo

Projeto Final de Curso apresentado ao Curso de Graduação em Engenharia Elétrica do Instituto Federal de Educação, Ciência e Tecnologia da Bahia, *campus* Vitória da Conquista, como requisito parcial para obtenção do título de Bacharel em Engenharia Elétrica.

Orientador: Dr. José Alberto Díaz Amado

Vitória da Conquista - BA

13 de agosto de 2024

FICHA CATALOGRÁFICA ELABORADA PELO SISTEMA DE BIBLIOTECAS DO IFBA, COM OS DADOS FORNECIDOS PELO(A) AUTOR(A)

S586c Silva, Jeferson Caio Oliveira

Controle adaptativo de tráfego: um estudo comparativo entre aprendizado por reforço e semáforos de tempo fixo./ Jeferson Caio Oliveira Silva; orientador José Alberto Díaz Amado -- Vitória da Conquista : IFBA, 2024.

68 p.

Trabalho de Conclusão de Curso (Engenharia Elétrica) -- Instituto Federal da Bahia, 2024.

1. Semáforos inteligentes. 2. Inteligência Artificial. 3. Treinamento por reforço. I. Díaz Amado, José Alberto, orient. II. TÍTULO.

CDD: 006.3

MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA BAHIA

ATA

CONTROLE ADAPTATIVO DE TRÁFEGO: UM ESTUDO COMPARATIVO ENTRE APRENDIZADO POR REFORÇO E SEMÁFOROS DE TEMPO FIXO

JEFERSON CAIO OLIVEIRA SILVA

A presente monografia de Projeto Final de Curso (PFC), apresentada em sessão realizada no dia 13 de agosto de 2024, foi avaliada como adequada para a obtenção do Grau de Bacharel em Engenharia Elétrica, julgada **APROVADA** em sua forma final pela Coordenação do Curso de Engenharia Elétrica do Instituto Federal de Educação, Ciência e Tecnologia da Bahia, *campus* Vitoria da Conquista.

BANCA EXAMINADORA

Prof. Dr. José Alberto Díaz Amado (Orientador)..... IFBA
Prof. Dr. Luis Paulo da Silva Carvalho ..... IFBA
Prof. Dr. Marcelo Mendonça Dos Santos ..... IFBA

Vitória da Conquista - Bahia

Documento assinado eletronicamente por **JOSE ALBERTO DIAZ AMADO**, **Professor Efetivo**, em 11/09/2024, às 12:19, conforme decreto nº 8.539/2015.

Documento assinado eletronicamente por **MARCELO MENDONCA DOS SANTOS**, **Professor(a) do Ensino Básico, Técnico e Tecnológico - EBTT**, em 11/09/2024, às 12:39, conforme decreto nº 8.539/2015.

Documento assinado eletronicamente por **LUIS PAULO DA SILVA CARVALHO**, **Professor Efetivo**, em 11/09/2024, às 12:51, conforme decreto nº 8.539/2015.

A autenticidade do documento pode ser conferida no site http://sei.ifba.edu.br/sei/controlador_externo.php?acao=documento_conferir&acao_origem=documento_conferir&id_orgao_acesso_externo=0 informando o código verificador **3664248** e o código CRC **01AE3963**.

AGRADECIMENTOS

Agradeço primeiramente a Deus, que em todo este tempo guiou os meus caminhos. Agradeço à minha família, sem o apoio e suporte deles eu não conseguiria. Agradeço ao meu orientador, José, cuja paciência sem limites permitiu que eu conseguisse terminar este trabalho. Obrigado a todos!

RESUMO

Este trabalho compara semáforos inteligentes, controlados por aprendizado por reforço, com semáforos de tempo fixo conforme o Manual Brasileiro de Sinalização de Trânsito, visando avaliar a eficácia na melhoria do fluxo de tráfego e na redução de congestionamentos do Brasil.

Foram implementados algoritmos de *Q-learning* e *Deep Q-Network* (DQN) em uma interseção simples e em uma rede com três semáforos alinhados. As simulações no ambiente SUMO compararam o desempenho dos semáforos inteligentes e dos semáforos de tempo fixo em termos de tempo de espera, número de paradas e velocidade média dos veículos.

Os resultados indicaram que, na interseção simples, o sistema DQN apresentou uma redução significativa no tempo total de espera, alcançando uma média de 39.24 segundos, representando uma redução de 58.8% em relação ao sistema de tempo fixo (95.29 segundos). O *Q-learning* teve uma média de 66.61 segundos. Quanto ao número de paradas, o DQN também obteve melhor desempenho, com uma média de 7.01 paradas, uma redução de 24.86% em comparação com o sistema de tempo fixo (9.33 paradas). O *Q-learning* apresentou 8.19 paradas.

Na rede com três semáforos alinhados, o DQN registrou o menor tempo total de espera, com 76.94 segundos, em comparação com o sistema de tempo fixo (94.94 segundos) e o *Q-learning* (84.97 segundos). Em termos de número de paradas, o DQN obteve a menor média, com 9.74 paradas, uma redução de 20.1% em comparação com o sistema de tempo fixo (12.20 paradas). O *Q-learning* teve uma média de 10.96 paradas.

Assim, a implementação de semáforos inteligentes com aprendizado por reforço pode melhorar a gestão de tráfego em ambientes urbanos.

Palavras-chave: Semáforos inteligentes, aprendizado por reforço, Q-learning, DQN, gerenciamento de tráfego.

ABSTRACT

This work compares intelligent traffic lights, controlled by reinforcement learning, with fixed-time traffic lights according to the Brazilian Traffic Signaling Manual, aiming to evaluate their effectiveness in improving traffic flow and reducing congestion in Brazil.

Q-learning and Deep Q-Network (DQN) algorithms were implemented in a simple intersection and a network with three aligned traffic lights. Simulations in the SUMO environment compared the performance of intelligent traffic lights and fixed-time traffic lights in terms of waiting time, number of stops, and average vehicle speed.

The results indicated that, in the simple intersection, the DQN system presented a 25% reduction in total stops (7.01 stops) compared to the fixed-time system (9.33 stops). Q-learning presented 8.19 stops. In terms of total waiting time, DQN achieved an average of 39.24 seconds, representing a 58.8% reduction compared to the fixed-time system (95.29 seconds). Q-learning had an average of 66.61 seconds.

In the network with three aligned traffic lights, DQN obtained the lowest average total stops (9.74 stops), a reduction of 20.1% compared to the fixed-time system (12.20 stops). Q-learning had an average of 10.96 stops. The total waiting time was lower in DQN (76.94 seconds) compared to the fixed-time system (94.94 seconds) and Q-learning (84.97 seconds).

Thus, the implementation of intelligent traffic lights with reinforcement learning can improve traffic management in urban environments.

Keywords: Intelligent traffic lights, reinforcement learning, Q-learning, DQN, traffic management.

Lista de Figuras

Lista de Tabelas

Lista de Códigos

Glossário: Símbolos e Siglas

Notação	Descrição	Páginas
A3C	Ator-Crítico de Vantagem Assíncrona (*Asynchronous Advantage Actor-Critic*)	22
COEEL	*Coordenação do Curso de Engenharia Elétrica do IFBA campus Vitória da Conquista*	i
CPU	*Central Processing Unit*	19
DENATRAN	Departamento Nacional de Trânsito	31
DNN	Deep Neural Networks	15, 16
DQN	Deep Q-Network	vi, 2, 16, 18, 22, 43, 47, 50, 52, 62
DRL	Deep Reinforcement Learning	17
DSRC	*Dedicated Short Range Communication (Comunicação Dedicada de Curto Alcance)*.	59
GPU	Unidade de Processamento Gráfico (*Graphics Processing Unit*)	19, 23
IPEA	Instituto de Pesquisa Econômica Aplicada	1

Notação	Descrição	Páginas
MDP	Markov Decision Process	6
MLP	Multi-Layer Perceptron	15
MV	Movimento	28
PPO	Otimização de Políticas Proximais (*Proximal Policy Optimization*)	22
ReLU	Rectified Linear Unit	14, 43
RNA	Rede de Neurônios Artificiais	14, 15
SUMO	Simulation of Urban MObility	2, 18, 20, 32, 62
TRACI	Traffic Control Interface	20, 21, 23, 40

Sumário

Capítulo 1

Introdução

Segundo dados do Instituto de Pesquisa Econômica Aplicada (IPEA), o crescimento acelerado das cidades brasileiras, juntamente com o aumento significativo do número de veículos, tem gerado desafios consideráveis para a gestão do tráfego urbano. Este cenário de superlotação das vias públicas não só provoca atrasos e frustrações para os cidadãos, como também contribui para o aumento da poluição ambiental e do consumo de combustíveis fósseis. A falta de políticas específicas que promovam alternativas de transporte eficientes e sustentáveis intensifica o uso do transporte individual, exacerbando problemas como acidentes de trânsito e a poluição. Nesse contexto, a eficiência dos sistemas semafóricos desempenha um papel crucial na melhoria do fluxo de tráfego e na mitigação dos impactos negativos associados aos congestionamentos, sendo essencial para a sustentabilidade urbana (Instituto de Pesquisa Econômica Aplicada, 2020).

Neste trabalho, é realizada uma comparação entre semáforos inteligentes, controlados por algoritmos de aprendizado por reforço (*Q-learning* e *Deep Q-Learning*), e semáforos de tempo fixo tradicionais em redes de tráfego, conforme especificado no Manual Brasileiro de Sinalização de Trânsito – Volume V: Sinalização Semafórica (DENATRAN, 2020). A abordagem de aprendizado por reforço permite que os semáforos aprendam e se adaptem às condições de tráfego em tempo real, potencialmente oferecendo uma solução mais dinâmica e eficiente em comparação com os métodos tradicionais.

1.1 Objetivo Geral

O objetivo geral deste trabalho é avaliar a eficácia de semáforos inteligentes, controlados por algoritmos de aprendizado por reforço, em comparação com semáforos de tempo fixo tradicionais, conforme especificado no manual brasileiro de semáforos, visando melhorar o fluxo de tráfego e reduzir o tempo de espera dos veículos.

1.2 Objetivos Específicos

Os objetivos específicos deste trabalho são:

1) Desenvolver e implementar algoritmos de aprendizado por reforço, especificamente *Q-learning e Deep Q-Network* (DQN), para o controle de semáforos;

2) Simular o desempenho dos semáforos inteligentes em uma interseção simples e em uma rede com três semáforos alinhados horizontalmente utilizando o ambiente *Simulation of Urban MObility* (SUMO);

3) Comparar os resultados dos semáforos inteligentes com os semáforos de tempo fixo, conforme o manual brasileiro de semáforos, em termos de tempo de espera, número de paradas e velocidade média dos veículos;

4) Analisar as vantagens e limitações dos semáforos inteligentes com aprendizado por reforço em relação aos semáforos de tempo fixo.

1.3 Justificativa

A escolha deste tema surge da necessidade urgente de melhorar o controle do tráfego nas cidades, onde o número de veículos cresce constantemente, pressionando as infraestruturas já sobrecarregadas. Os semáforos tradicionais, com tempos fixos, não conseguem se adaptar às mudanças rápidas nas condições de tráfego. Em contraste, os semáforos inteligentes, que utilizam algoritmos de aprendizado por reforço, podem ajustar os tempos de sinalização em tempo real, oferecendo uma resposta mais eficaz e ágil às necessidades das vias (TAYLOR; STONE,

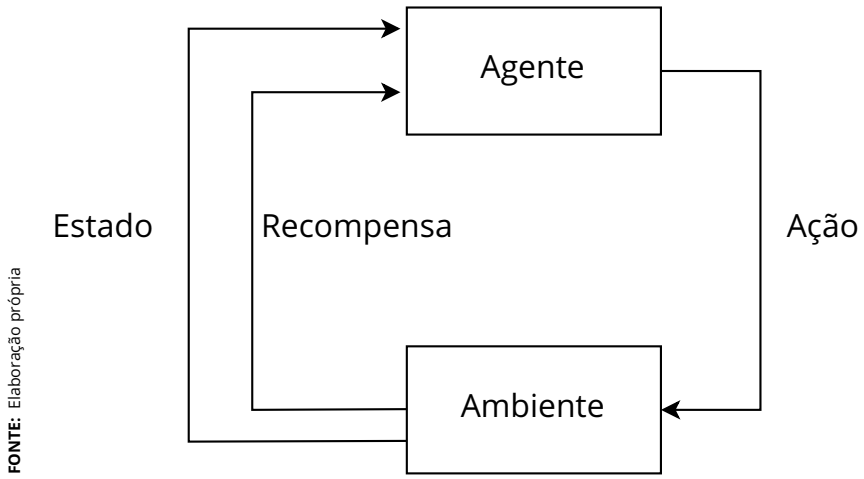
2016).

Capítulo 2

Fundamentação Teórica

2.1 Aprendizado por Reforço

O aprendizado por reforço é uma área de aprendizado de máquina onde um agente aprende a tomar decisões ao interagir com um ambiente dinâmico. Segundo (SUTTON; BARTO, 2018), os principais componentes da aprendizagem por reforço são o ambiente, o agente, as ações, os estados e as recompensas. A Figura 2.1 ilustra este processo:



FONTE: Elaboração própria

Figura 2.1 – *Diagrama aprendizado por reforço*

No contexto do aprendizado por reforço, o processo de decisão do agente é dividido em várias iterações conhecidas como *steps*. Cada step é uma unidade de interação entre o agente e o ambiente. Durante um step, o agente realiza um ciclo no qual:

1) **Percebe o estado atual** do ambiente;
2) **Escolhe uma ação** com base em sua política ou estratégia;
3) **Executa a ação** no ambiente;
4) **Recebe um *feedback*** (recompensa) e o novo estado resultante dessa ação.

O **ambiente**, segundo (ZHANG et al., 2019), é o contexto no qual o agente opera e responde às suas ações, proporcionando novos estados e recompensas. Ele é externo ao agente e inclui tudo com o que o agente pode interagir. As características e dinâmicas do ambiente determinam como o agente aprende e se adapta ao longo do tempo, influenciando diretamente o processo de tomada de decisão e aprendizado contínuo do agente dentro do ambiente dinâmico.

O **agente**, por sua vez, é a entidade que toma decisões e aprende com as consequências de suas ações. Ele observa o estado atual do ambiente, escolhe e executa ações e recebe *feedback* na forma de recompensas. (KAELBLING; LITTMAN; MOORE, 1996) define o agente como o componente que possui a capacidade de perceber o ambiente através de sensores e influenciar o ambiente através de atuadores, fazendo com que a aprendizagem ocorra através da interação contínua com o ambiente.

As **ações** são as possíveis decisões que o agente pode tomar em cada estado. Elas representam as escolhas que o agente tem disponíveis para tentar influenciar o ambiente de maneira a maximizar suas recompensas futuras. (SUTTON; BARTO, 2018) explica que as ações são fundamentais porque elas determinam a trajetória que o agente seguirá através dos estados, impactando diretamente no processo de aprendizado.

Os **estados** são representações do ambiente em diferentes momentos. Cada estado encapsula toda a informação relevante que o agente precisa para tomar uma decisão informada. Segundo (ZHANG et al., 2019), um estado é uma descrição completa da situação atual do ambiente, e a escolha apropriada dos estados é crucial para o sucesso do agente.

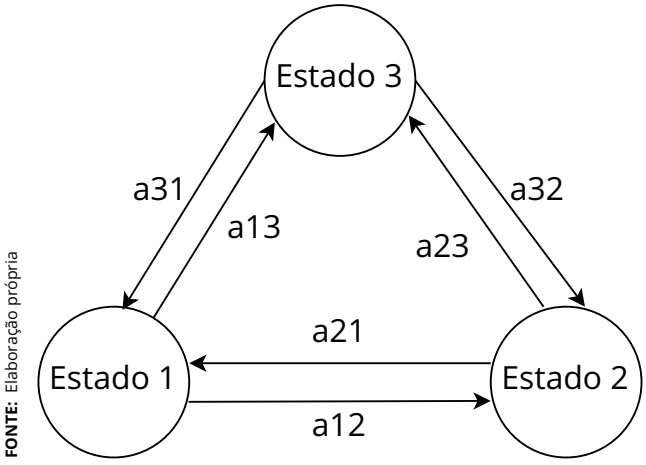
As **recompensas** são *feedbacks* numéricos que indicam a qualidade das ações tomadas pelo agente. Elas fornecem a informação necessária para que o agente aprenda quais ações são mais vantajosas em termos de maximizar algum critério de desempenho ao longo do tempo. (SUTTON; BARTO, 2018) afirma que as re-

compensas são a única forma de *feedback* que o agente utiliza para ajustar seu comportamento e aprender a política ótima.

2.2 Processos de Decisão de Markov (MDPs)

Os Processos de Decisão de Markov (MDP) são uma estrutura matemática usada para modelar decisões sequenciais em ambientes estocásticos, onde os resultados dependem tanto do acaso quanto do controle de um agente decisor, conforme discutido por (PUTERMAN, 2014). Um MDP é composto por um conjunto de estados S, um conjunto de ações A, uma função de transição de estados P, que define a probabilidade de transição de um estado s para outro estado s' dado uma ação a, e uma função de recompensa R, que especifica a recompensa após essa transição.

Para ilustrar essa definição, considere o diagrama mostrado na Figura 2.2, com três estados que representam diferentes condições do sistema. As transições entre esses estados são indicadas pelas setas rotuladas com ações, como a_{12}, a_{23}, e a_{31}, que representam as possíveis decisões que o agente pode tomar. A função de transição T determina a probabilidade de o sistema se mover de um estado para outro, dado que uma ação específica foi executada. Por exemplo, a probabilidade de transitar do Estado 1 para o Estado 2 após executar a ação a_{12} é dada por $T(\text{Estado 1}, a_{12}, \text{Estado 2})$. Da mesma forma, a função de recompensa R atribui um valor a cada transição. Assim, o valor da recompensa associado a mover-se do Estado 1 para o Estado 2 utilizando a ação a_{12} é representado por $R(\text{Estado 1}, a_{12}, \text{Estado 2})$, influenciando, assim, a escolha das ações pelo agente.



FONTE: Elaboração própria

Figura 2.2 – *Diagrama do processo de decisão de Markov*

Os processos de Markov incluem duas propriedades fundamentais. A primeira é a **falta de memória**, que significa que o estado futuro de um sistema depende apenas do estado atual, não dos eventos passados. A segunda é a **probabilidade de transição**, onde a transição para um estado futuro é determinada unicamente pelo estado presente.

2.3 Função Valor-Ação

A Função Valor-Ação, $Q(s,a)$, mostrada na Equação 2.1, é uma função que estima o valor esperado da recompensa acumulada que pode ser obtida a partir de um estado s ao se tomar uma ação a. Em termos mais simples, a função valor-ação mede a qualidade de uma ação específica em um estado específico, ajudando o agente a decidir quais ações tomar para obter a maior recompensa possível no futuro (SUTTON; BARTO, 2018). Formalmente, a função valor-ação é definida como:

$$Q(s,a) = \mathbb{E}[R_t | s_t = s, a_t = a] \tag{2.1}$$

onde:

- $Q(s,a)$: Função valor-ação para o estado s e a ação a.
- $\mathbb{E}$: Valor esperado.
- R_t: Recompensa recebida no tempo t.
- s_t: Estado no tempo t.
- a_t: Ação tomada no tempo t.

Isso significa que $Q(s,a)$ é a expectativa da recompensa total que o agente pode obter ao começar no estado s, tomar a ação a e seguir a melhor estratégia possível daí em diante.

2.4 Políticas

Uma política, π, é uma regra ou estratégia que define o comportamento do agente, especificando a ação a que o agente deve tomar em um estado s. A política pode ser vista como um mapa que direciona o agente em cada situação possível (KAELBLING; LITTMAN; MOORE, 1996). Em termos matemáticos, uma política pode ser representada como $\pi(a|s)$, na Equação 2.2, que determina a probabilidade de selecionar cada ação a dado o estado s:

$$\pi(a|s) = P(a|s) \tag{2.2}$$

onde:

- $\pi(a|s)$: Política que mapeia o estado s para a ação a.
- $P(a|s)$: Probabilidade de selecionar a ação a dado o estado s.

Uma política é dita ótima se maximiza o valor esperado da recompensa acumulada ao longo do tempo. O objetivo da aprendizagem por reforço é encontrar uma política ótima, π^*, que permita ao agente obter a maior recompensa possível a partir de qualquer estado.

2.5 Tabela Q

A Tabela Q é uma representação tabular da Função Valor-Ação $Q(s, a)$. Em ambientes discretos, onde o número de estados e ações é finito, o agente pode armazenar os valores Q em uma tabela, onde cada linha corresponde a um estado s e cada coluna corresponde a uma ação a. Durante o processo de aprendizagem, o agente atualiza os valores na Tabela Q para refletir a estimativa atualizada da recompensa esperada (WATKINS; DAYAN, 1992). A tabela Q é usada em algoritmos como o Q-Learning, onde o agente aprende iterativamente a partir de suas interações com o ambiente.

A Tabela 2.1 apresenta um exemplo de Q-table, ilustrando a relação entre estados e ações com seus respectivos valores Q.

Tabela 2.1 – *Q-table*

Estado	Ação 1	Ação 2	...	Ação N
Estado 1	Valor Q11	Valor Q12	...	Valor Q1N
Estado 2	Valor Q21	Valor Q22	...	Valor Q2N
Estado 3	Valor Q31	Valor Q32	...	Valor Q3N
...	...	...	...	...
Estado M	Valor QM1	Valor QM2	...	Valor QMN

FONTE: Elaboração própria.

2.6 Equação de Bellman

A Equação de Bellman é uma relação de recorrência que fornece uma maneira de atualizar a Função Valor-Ação com base nas recompensas recebidas e nas estimativas futuras. A Equação de Bellman para a Função Valor-Ação $Q(s, a)$ é dada pela Equação 2.3:

$$Q(s, a) = R(s, a) + \gamma \sum_{s'} P(s'|s, a) \max_{a'} Q(s', a') \tag{2.3}$$

onde:

- $Q(s, a)$: Função valor-ação para o estado s e a ação a.
- $R(s, a)$: Recompensa imediata recebida ao tomar a ação a no estado s.
- γ: Fator de desconto que valoriza recompensas futuras.
- $P(s'|s, a)$: Probabilidade de transição para o estado s' dado o estado atual s e a ação a.
- $\max_{a'} Q(s', a')$: Máximo valor de Q para todas as ações a' no novo estado s'.

A Equação de Bellman expressa a relação entre o valor atual e o valor futuro esperado, permitindo que o agente ajuste suas estimativas de Q iterativamente (WATKINS; DAYAN, 1992). A capacidade de um agente de ajustar suas estimativas de valor-ação com base em novas informações e experiências permite a aprendizagem e a melhoria contínua de seu desempenho.

2.7 Estratégia Epsilon-Greedy

A estratégia *epsilon-greedy* é utilizada na aprendizagem por reforço para balancear *exploration* e *exploitation* na tomada de decisões. *Exploration* refere-se à tentativa de novas ações para descobrir suas recompensas potenciais, enquanto *exploitation* envolve a seleção de ações que já são conhecidas por oferecer as melhores recompensas.

No contexto do controle de semáforos, essa estratégia determina se o agente deve escolher uma ação aleatória (*exploration*) ou a ação com maior valor Q (*exploitation*). Com probabilidade ϵ, o agente escolhe uma ação aleatória, e com probabilidade $1 - \epsilon$, escolhe a ação que maximiza o valor da função Q.

Código 2.1 – *Estratégia epsilon-greedy*

```
Para cada episódio:
    Se número_aleatorio < e:
        Selecionar ação aleatória
    Se não:
        Selecionar acao com maior valor Q
```

Por exemplo, com $\epsilon = 0.1$, há 10% de chance de o agente escolher aleatoriamente uma ação, e 90% de chance de escolher a ação que maximiza o valor Q, otimizando o fluxo de tráfego na interseção.

2.8 Algoritmo de Q-learning

O *Q-learning* é um algoritmo *off-policy*, ou seja, aprende a política ótima independentemente da política que está sendo seguida pelo agente para coletar dados. Em outras palavras, a política usada para tomar decisões (política de comportamento) pode ser diferente da política que está sendo aprendida (política alvo). O *Q-learning* aprende a função Q através de atualizações iterativas,Em que a Equação 2.4 realiza a atualização (WATKINS; DAYAN, 1992):

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \max_{a'} Q(s',a') - Q(s,a) \right] \tag{2.4}$$

onde:

- α é a taxa de aprendizado,
- r é a recompensa recebida após a ação a,
- γ é o fator de desconto,
- s é o estado atual,
- a é a ação tomada no estado s,
- s' é o novo estado resultante após a ação a,
- $\max_{a'} Q(s', a')$ é o valor máximo da função Q para o novo estado s' considerando todas as possíveis ações a'.

Esta atualização ajusta a estimativa de $Q(s, a)$ com base na diferença entre a recompensa observada mais o valor descontado da melhor ação subsequente e a estimativa atual de $Q(s, a)$ (WATKINS; DAYAN, 1992).

O pseudocódigo para o algoritmo Q-learning pode ser descrito no Código 2.2:

Código 2.2 – *Pseudocodigo do Algoritmo Q-learning*

```
1. Inicialize Q(s, a) arbitrariamente
2. Para cada episodio:
    a. Inicialize s
    b. Para cada passo do episodio:
        i. Escolha a de s usando uma politica (e-greedy)
        ii. Execute a, observe r e s'
        iii. Atualize Q(s, a) usando a formula de atualizacao
        iv. s <- s'
        v. Se s for terminal, encerre o episodio
```

No Aprendizado por Reforço, *exploration versus exploitation* é um dilema central (AUER; CESA-BIANCHI; FISCHER, 2002). A estratégia ϵ-*greedy* é uma abordagem comum onde, com probabilidade ϵ, uma ação aleatória é escolhida (*exploration*), e com probabilidade $1 - \epsilon$, a melhor ação conhecida é escolhida (*exploitation*). Ajustar ϵ ao longo do tempo permite balancear a descoberta de novas estratégias e a utilização das melhores estratégias conhecidas (CHEN; WANG et al., 2022).

Os parâmetros de exploração, como o valor inicial de ϵ, a taxa de decaimento e o valor mínimo de ϵ, são críticos para o desempenho do algoritmo (CHEN; WANG et al., 2022). Em sistemas de semáforos inteligentes, esses parâmetros precisam ser cuidadosamente ajustados para garantir que o sistema possa descobrir

e adaptar-se a novas condições de tráfego sem se tornar excessivamente conservador.

O *Q-learning* tem se mostrado uma abordagem eficiente para o controle de semáforos inteligentes, mas enfrenta desafios consideráveis quando aplicado a cenários mais complexos, como os ambientes urbanos. Um desses desafios é lidar com o grande número de estados possíveis, que muitas vezes variam continuamente. Em cidades grandes, o tráfego muda de maneira fluida — as variáveis como a posição e a velocidade dos veículos estão em constante transformação, o que caracteriza os chamados estados contínuos. Esses estados não podem ser facilmente representados em tabelas Q, que são mais adequadas para ambientes discretos. Isso cria um problema de escalabilidade, já que a tabela Q cresce exponencialmente conforme aumentam os estados e ações, resultando em maior consumo de memória e maior tempo de processamento (MELO, 2001). Por isso, o *Q-learning* tradicional pode se tornar impraticável em ambientes de larga escala, como o controle de tráfego em áreas urbanas densas. Para superar essas limitações, foram desenvolvidas novas abordagens, como o *Deep Q-Learning*, que utiliza redes neurais para simplificar o processo de tomada de decisão em espaços de estado muito grandes ou contínuos. Essa técnica ajuda a generalizar o aprendizado, tornando-o viável para aplicações mais complexas, como o gerenciamento eficiente de semáforos em grandes cidades (MNIH et al., 2015a).

2.9 Redes Neurais

Redes neurais são modelos computacionais inspirados no funcionamento do cérebro humano, capazes de reconhecer padrões e aprender a partir de dados. São compostas por unidades de processamento chamadas neurônios artificiais, que estão organizados em camadas: a camada de entrada, camadas escondidas (ou ocultas) e a camada de saída. Cada neurônio recebe entradas, realiza uma operação matemática sobre elas (geralmente uma soma ponderada seguida de uma função de ativação) e passa a saída para os neurônios da próxima camada (LECUN; BENGIO; HINTON, 2015).

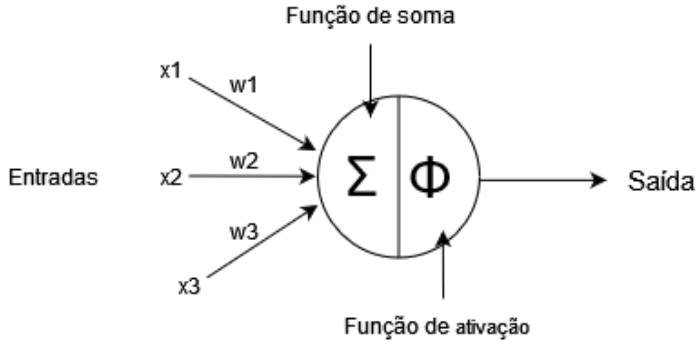
2.9.1 Neurônios Artificiais

Os neurônios artificiais são as unidades básicas das redes neurais, inspiradas nos neurônios biológicos. Cada neurônio realiza uma operação matemática sobre suas entradas. O neurônio recebe múltiplas entradas ($x_1, x_2, \ldots, x_n$), aplica um peso (w) a cada entrada, soma esses valores e adiciona um bias (b). A saída (z), Equação 2.5 do neurônio é então passada por uma função de ativação (ϕ), Equação 2.6 (GROSSI; BUSCEMA, 2008).

$$z = \sum_{i=1}^{n} w_i x_i + b \tag{2.5}$$

$$a = \phi(z) \tag{2.6}$$

A Figura 2.3 mostra a representação de Neurônio artificial.



FONTE: Elaboração própria.

Figura 2.3 – *Representação de neurônios artificiais*

As redes neurais artificiais são constituídas por diversos neurônios organizados em camadas. Essas camadas incluem uma camada de entrada, uma ou mais camadas ocultas e uma camada de saída. A definição e os componentes de uma RNA envolvem, além dos neurônios, as funções de ativação, os métodos de treinamento, os algoritmos de otimização e os critérios de erro. Cada neurônio em uma camada está conectado a neurônios da camada subsequente, criando uma rede densa de conexões que permite a propagação de informações e a realização de cálculos complexos (GROSSI; BUSCEMA, 2008).

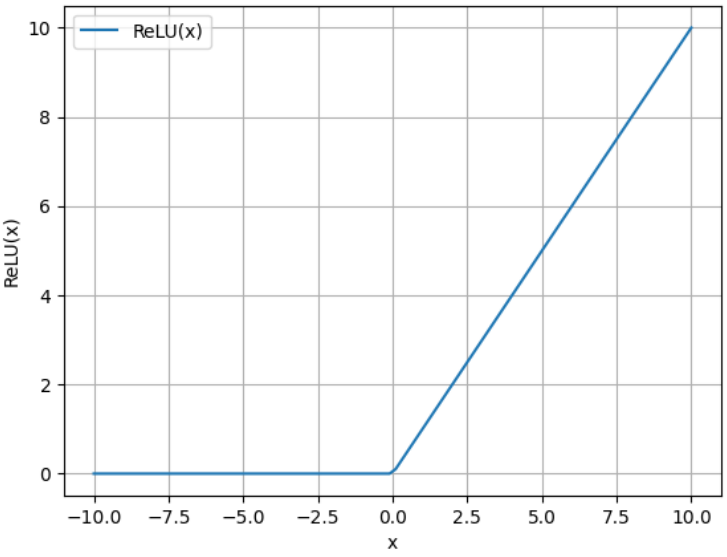
2.9.2 Função de Ativação

A função de ativação é uma componente nas RNAs que introduz não-linearidades no modelo (IN, 2023), permitindo a aprendizagem de padrões complexos. Uma das funções de ativação comuns é a ReLU (*Rectified Linear Unit*), que é definida pela Equação 2.7 :

$$\text{ReLU}(x) = \max(0, x) \tag{2.7}$$

A função ReLU é utilizada devido à sua simplicidade e eficiência computacional. Ela resolve o problema de saturação de gradientes encontrado em outras funções de ativação, como a sigmoide e a tangente hiperbólica, embora possa sofrer do problema de "neurônios mortos", onde alguns neurônios podem parar de aprender se a entrada for sempre negativa.

A Figura 2.4 ilustra a forma da função ReLU, onde é possível observar que todos os valores negativos são mapeados para zero, enquanto os valores positivos são mantidos inalterados. Isso cria uma linearidade por partes que permite a introdução de não-linearidade no modelo sem saturar os gradientes.



FONTE: Elaboração própria.

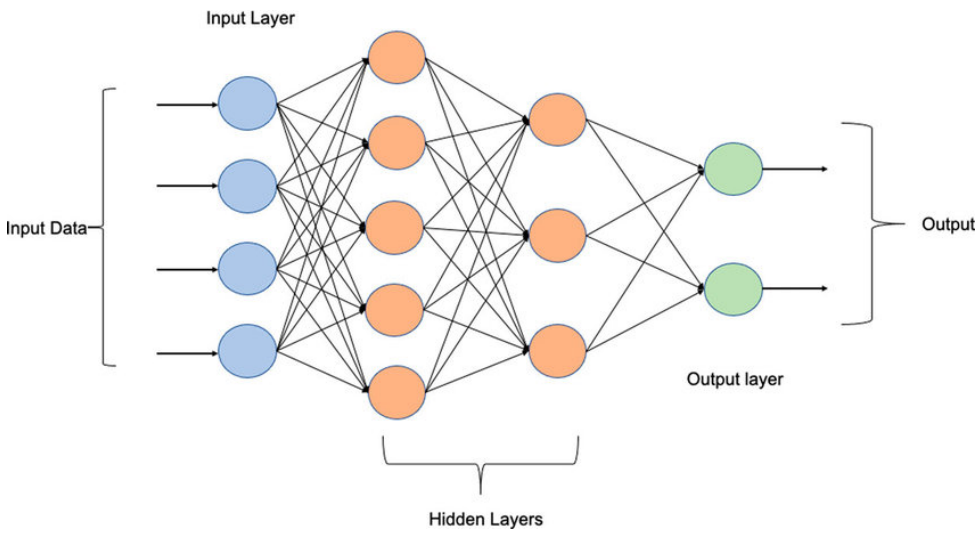
Figura 2.4 – *Gráfico da função de ativação ReLU.*

2.9.3 Perceptron

O *Perceptron* é a forma mais simples de uma Rede de Neurônios Artificiais RNA, composta por um único neurônio que faz uma classificação binária. Ele realiza uma combinação linear das entradas, seguida por uma função de ativação, e é treinado utilizando o algoritmo de aprendizado do perceptron. Matematicamente, o *perceptron* pode ser expresso como:

$$y = \begin{cases} 1 & \text{se } \sum_{i=1}^{n} w_i x_i + b > 0 \\ 0 & \text{caso contrário} \end{cases}$$

O Perceptron Multicamadas (MLP), por sua vez, é uma generalização do perceptron que inclui múltiplas camadas de neurônios. Essa estrutura é capaz de resolver problemas de classificação não-linear e de aproximar funções complexas (MOHAMMADZADEH et al., 2022). O treinamento do MLP é realizado através do algoritmo de retropropagação, que ajusta os pesos e *biases* da rede.



FONTE: (MOHAMMADZADEH et al., 2022) Rede Neural Perceptron Multicamadas.

Figura 2.5 – *Arquitetura Básica de uma Rede Neural Perceptron Multicamadas (MLP)*

2.10 Rede Neural Profunda

Redes neurais profundas (*Deep Neural Networks*, DNN) são uma classe de modelos de aprendizado de máquina compostos por múltiplas camadas de neurônios

artificiais. Essas camadas são projetadas para aprender representações hierárquicas dos dados de entrada, permitindo a detecção e modelagem de padrões complexos. DNNs são amplamente aplicadas em diversas áreas, incluindo reconhecimento de imagem, processamento de linguagem natural e controle de tráfego (GOODFELLOW; BENGIO; COURVILLE, 2016).

No contexto deste trabalho, uma aplicação eficaz das DNNs é o Deep Q-Network (DQN). O DQN é uma técnica de aprendizado por reforço que combina redes neurais profundas com o algoritmo de *Q-Learning*, permitindo que o sistema aprenda políticas de controle diretamente a partir de dados de entrada de alto nível, como imagens ou dados de sensores de tráfego (MNIH et al., 2015a).

As principais técnicas utilizadas pelo DQN incluem o *Replay* de Experiência e as Redes de Alvo. A técnica de *Replay* de Experiência consiste em armazenar transições de estado, ação, recompensa e próximo estado em uma memória de *replay*. Durante o treinamento, amostras aleatórias dessa memória são utilizadas para atualizar a rede neural, o que ajuda a estabilizar o treinamento ao quebrar a correlação entre sequências consecutivas de dados (LIN, 1992).

Por outro lado, as Redes de Alvo são cópias da rede principal que são atualizadas periodicamente. Essas redes fornecem valores de Q mais estáveis para a atualização do treinamento, reduzindo a variação e, consequentemente, melhorando a estabilidade do modelo.

Capítulo 3

Estado da Arte

Para elaboração do semáforo de tempo fixo, foi adotado o Manual Brasileiro de Sinalização de Trânsito – Volume V: Sinalização Semafórica (DENATRAN, 2020), que fornece diretrizes para a configuração e operação de sistemas semafóricos no Brasil. Apesar da importância deste manual, ele não aborda a utilização de técnicas avançadas de controle de tráfego, como o aprendizado por reforço.

Além disso, artigos e estudos sobre otimização de semáforos por aprendizagem por reforço e outros tipos de algoritmos foram referenciados para contextualizar este trabalho:

- *Adaptive Traffic Signal Control with Deep Reinforcement Learning*: Este estudo apresenta um modelo de controle de sinal de tráfego utilizando DRL, mostrando reduções significativas nos atrasos de tráfego. No entanto, falta uma análise detalhada sobre a aplicação em cenários específicos como os brasileiros (MURESAN; FU; PAN, 2019).
- *Optimization of Traffic Light Control Using Fuzzy Logic Sugeno Method*: Explora a aplicação de lógica *fuzzy* para otimizar o controle de sinais de tráfego, destacando a integração com métodos de aprendizado por reforço. Contudo, este trabalho não compara diretamente com semáforos de tempo fixo conforme normas específicas (KARTIKASARI; PRAKARSA; PRADEKA, 2020).
- *Traffic Signal Optimization Control Method Based on Adaptive Weighted Averaged Double Deep Q Network*: Descreve um sistema de DRL para otimização de sinais de tráfego que reduz o tempo médio de espera dos veí-

culos. Este estudo se concentra em melhorias específicas de algoritmos, mas não aborda a comparação com sistemas de tempo fixo tradicionais (CHEN et al., 2023).

- *Distributed Signal Control of Arterial Corridors Using Multi-Agent Deep Reinforcement Learning*: Examina a utilização de DRL em um ambiente multi-agente para a coordenação de sinais de tráfego em tempo real. Embora inovador, carece de uma análise comparativa detalhada com sistemas fixos e a aplicabilidade em diferentes contextos de infraestrutura (LI et al., 2023).
- *Transfer Learning in Deep Reinforcement Learning: A Survey*: Analisa como o aprendizado por transferência pode ser aplicado em DRL para melhorar o desempenho de modelos em diferentes cenários de controle de tráfego. Este estudo destaca a eficiência do aprendizado por transferência, mas não considera a implementação prática em sistemas semafóricos existentes (ZHU et al., 2023).

Em contraste com esses estudos, o presente trabalho se diferencia ao focar na comparação direta entre algoritmos de aprendizado por reforço (*Q-learning e DQN*) e semáforos de tempo fixo, especificamente no contexto das normas brasileiras de sinalização de trânsito. A análise realizada em dois cenários distintos, utilizando o ambiente SUMO, oferece uma base robusta para entender as vantagens e limitações de cada abordagem em condições específicas de tráfego. Este estudo visa preencher a lacuna existente na literatura sobre a aplicação prática e comparativa de técnicas avançadas de controle de tráfego em conformidade com as normas brasileiras.

Capítulo 4

Ferramentas Utilizadas

Este capítulo apresenta as principais ferramentas e tecnologias empregadas no desenvolvimento do sistema de controle de tráfego com aprendizagem por reforço. Todo o desenvolvimento foi realizado utilizando a linguagem de programação *Python*, devido à sua flexibilidade e vasta coleção de bibliotecas para aprendizado de máquina e simulação. A seguir, são detalhados os programas e bibliotecas que foram utilizados para a modelagem, simulação e implementação dos algoritmos de controle de tráfego. Cada seção descreve uma ferramenta específica e como elas foram aplicadas neste projeto.

4.1 Hardware Utilizado

Para o desenvolvimento e treinamento das redes neurais utilizadas neste trabalho, foi empregado um notebook com as seguintes especificações de hardware:

- **Processador:** Intel(R) Core(TM) i5-4200U CPU @ 1.60GHz, 2.30 GHz, 2 núcleos, 4 threads.
- **Memória RAM:** 6,00 GB.
- **Sistema Operacional:** *Microsoft Windows 10 Home Single Language*.

Uma limitação importante deste hardware é a ausência de uma placa de vídeo dedicada (GPU), o que forçou a utilização da CPU para o treinamento das redes neurais. Isso impacta diretamente no tempo de treinamento, uma vez que

GPUs são mais adequadas para operações de processamento paralelo intensivo, comuns em tarefas de aprendizado profundo. No entanto, as configurações utilizadas foram suficientes para a realização dos experimentos propostos, com o devido ajuste nos parâmetros de treinamento para adequação à capacidade computacional disponível.

4.2 SUMO

O SUMO (*Simulation of Urban MObility*) é um simulador de tráfego multimodal e altamente eficiente que permite a modelagem e simulação de redes de tráfego urbano. Desenvolvido para fornecer uma plataforma flexível e extensível, o SUMO é amplamente utilizado em pesquisas e desenvolvimentos que envolvem a análise de sistemas de transporte e o impacto de diferentes estratégias de controle de tráfego (Eclipse Foundation, 2018).

O SUMO permite a criação de cenários realistas de tráfego, onde veículos, pedestres e transportes públicos podem ser simulados em um ambiente urbano detalhado. Os usuários podem definir redes de ruas complexas, configurar sinais de trânsito, e simular o comportamento do tráfego em diferentes condições. Essa capacidade de simulação detalhada é crucial para testar e validar algoritmos de controle de tráfego baseados em aprendizado por reforço, que dependem de dados realistas para otimizar suas políticas de decisão.

Além disso, o SUMO suporta a integração com outras ferramentas e bibliotecas através da interface TRACI (*Traffic Control Interface*). Essa interface permite a comunicação em tempo real entre o SUMO e programas externos, possibilitando que os agentes de aprendizado por reforço interajam diretamente com a simulação. Isso é fundamental para o treinamento de modelos, pois permite que os agentes executem ações e coletem dados em um ambiente dinâmico e interativo.

O SUMO também oferece uma variedade de funcionalidades adicionais, como a importação de mapas do *OpenStreetMap*, a configuração de rotas específicas para veículos, e a simulação de diferentes tipos de infraestruturas de transporte. Essas funcionalidades permitem a criação de simulações altamente customizadas e detalhadas, adequadas para uma ampla gama de estudos e experimentos no campo do gerenciamento de tráfego.

Outro aspecto importante do SUMO é sua capacidade de simulação em larga

escala, permitindo a modelagem de grandes áreas urbanas com milhares de veículos em movimento. Essa capacidade é essencial para avaliar o impacto de estratégias de controle de tráfego em redes de transporte complexas e densamente povoadas. A precisão e a escalabilidade do SUMO o tornam uma ferramenta indispensável para pesquisadores e engenheiros que buscam desenvolver soluções inovadoras para os desafios do tráfego urbano.

4.3 TRACI

A interface TRACI (Traffic Control Interface) permite a comunicação entre o SUMO e programas externos. Isso é essencial para implementar o controle dinâmico dos semáforos, permitindo que os agentes de aprendizado por reforço interajam com a simulação em tempo real. O uso do TRACI facilita a execução de ações e a coleta de dados durante a simulação, aspectos essenciais para o treinamento de modelos de aprendizado por reforço (Eclipse Foundation, 2024).

O TRACI fornece uma ampla gama de comandos que permitem o controle detalhado de diversos elementos da simulação, como a alteração dos estados dos semáforos, a manipulação do tráfego de veículos e a coleta de informações específicas sobre o fluxo de tráfego. Esse nível de controle e observação é crucial para a aplicação de técnicas de aprendizado por reforço, que dependem de interações contínuas com o ambiente para otimizar políticas de decisão.

Além disso, a interface TRACI suporta a execução de simulações distribuídas, onde múltiplos agentes de controle podem operar simultaneamente em diferentes partes da rede de tráfego. Isso permite a implementação de sistemas de controle descentralizados e colaborativos, que são essenciais para gerenciar redes de tráfego complexas e interconectadas. A flexibilidade do TRACI em integrar-se com diversas linguagens de programação, como *Python* e *C++*, amplia ainda mais seu potencial para pesquisas e desenvolvimento de soluções inovadoras no campo de gerenciamento de tráfego.

4.4 Gymnasium

O *Gymnasium* é um *toolkit* amplamente utilizado para o desenvolvimento e comparação de algoritmos de aprendizado por reforço. Ele fornece ambientes pa-

dronizados que facilitam o treinamento e a avaliação de diferentes algoritmos. A padronização e a flexibilidade do *Gymnasium* tornam-no ideal para o desenvolvimento de sistemas complexos de controle de tráfego, onde é necessário um ambiente que possa ser facilmente configurado e adaptado(Farama Foundation, 2024).

Um dos principais benefícios do *Gymnasium* é sua compatibilidade com diversos algoritmos de aprendizado por reforço, desde os mais básicos até os mais avançados, como DQN (*Deep Q-Network*), PPO *(Proximal Policy Optimization)* e A3C *(Asynchronous Advantage Actor-Critic)*. Essa compatibilidade permite que pesquisadores e desenvolvedores comparem o desempenho de diferentes abordagens de maneira sistemática.

Além disso, o *Gymnasium* oferece ferramentas para monitorar o progresso do treinamento dos agentes, incluindo métricas de desempenho e visualizações de políticas. Isso é fundamental para ajustar hiperparâmetros e melhorar a eficiência do aprendizado. A capacidade de criar ambientes personalizados no *Gymnasium* permite simular cenários de tráfego específicos e testar como diferentes políticas de controle afetam o fluxo de tráfego.

O *Gymnasium* também facilita a integração com bibliotecas de aprendizado profundo, como *TensorFlow* e *PyTorch*, permitindo a implementação de redes neurais complexas para a tomada de decisão. Essa integração é essencial para o desenvolvimento de agentes de aprendizado por reforço que possam lidar com a alta dimensionalidade e a complexidade dos dados de tráfego.

4.5 Stable Baselines3

Esta biblioteca oferece implementações estáveis e eficientes de diversos algoritmos de aprendizado por reforço. Utilizar o *Stable Baselines3* permite a integração de algoritmos avançados como o *Deep Q-Network* e *Q-Learning*, proporcionando uma base sólida para a experimentação e desenvolvimento de soluções de controle de tráfego (Raffin, Antonin and Hill, Ashley and Gleave, Adam and Kanervisto, Anssi and Ernestus, Maximilian and Dormann, Noah, 2024).

O *Stable Baselines3* é projetado para ser fácil de usar e bem documentado, o que acelera o processo de desenvolvimento e permite que pesquisadores e engenheiros foquem na experimentação e na melhoria dos algoritmos.

Uma característica importante do *Stable Baselines3* é sua capacidade de re-

alizar experimentos reproduzíveis. Isso é alcançado através da configuração consistente de sementes aleatórias e da estruturação sistemática dos experimentos. A reprodutibilidade é crucial em pesquisas científicas para validar resultados e compará-los de forma justa.

Além disso, o *Stable Baselines3* integra-se bem com o *Gymnasium*, facilitando a criação de pipelines de treinamento completos e eficientes. A biblioteca também suporta o uso de *GPU* para acelerar o treinamento de modelos complexos, o que é especialmente útil em simulações de tráfego onde a velocidade de processamento pode ser um gargalo.

O uso do *Stable Baselines3* em conjunto com outras ferramentas, como o TRACI e o *Gymnasium*, cria um ecossistema robusto para o desenvolvimento de sistemas inteligentes de controle de tráfego, permitindo a exploração de novas fronteiras no gerenciamento eficiente e adaptativo de redes de tráfego urbano.

Capítulo 5

Metodologia

Neste capítulo, será desenvolvido todo o processo de construção do semáforo fixo e dos algoritmos de aprendizagem por reforço, *Q-learning* e Deep Q-Network. Cada um desses modelos foi implementado e testado em um ambiente simulado no SUMO para avaliar sua eficiência na gestão do fluxo de tráfego.

5.1 Definição dos Ambientes

A escolha dos ambientes para o treinamento dos semáforos inteligentes buscou equilibrar realismo e viabilidade computacional. A interseção simples foi selecionada por oferecer um ambiente de controle direto e menor variabilidade, o que facilita a análise inicial dos algoritmos. Esse cenário permite observar o comportamento do algoritmo em uma situação típica, onde a otimização dos tempos de semáforo pode ser testada de maneira mais direta.

Por outro lado, o cenário com três interseções alinhadas foi escolhido para simular um ambiente urbano mais dinâmico, com múltiplas interações entre semáforos. Este cenário representa um desafio adicional, embora tenha sido limitado a três interseções para manter a complexidade dentro dos limites operacionais do *Q-Learning* e do hardware disponível. Além de ser um caso realista em termos de organização urbana, essa escolha permite testar a capacidade dos algoritmos de gerenciar coordenação de tráfego em vias conectadas, sem ultrapassar as capacidades computacionais.

5.2 Métricas de avaliação

Foram utilizadas três métricas principais de avaliação para medir o desempenho de ambos os métodos: o total de paradas, o tempo total de espera e a velocidade média do sistema.

1) **Total de Paradas**: Mede quantas vezes os veículos pararam na rede. O objetivo é reduzir esse número para melhorar o fluxo de tráfego.
2) **Tempo Total de Espera**: Representa o tempo acumulado em que os veículos ficaram parados nos semáforos. A redução dessa métrica indica maior eficiência do sistema.
3) **Velocidade Média do Sistema**: Indica a velocidade média dos veículos durante a simulação. Aumentar essa métrica reflete um tráfego mais fluido.

5.3 Descrição das Redes de Tráfego

As redes foram simplificadas e elaboradas especificamente para observação do trânsito de veículos, os fluxos de veículos foram definidos com base nos exemplos fornecidos pelo Manual Brasileiro de Sinalização de trânsito (DENATRAN, 2020). Foram considerados dois cenários distintos para a análise: uma interseção simples e uma rede com três semáforos alinhados horizontalmente.

5.3.1 Interseção Simples

A primeira rede de tráfego consiste em uma interseção simples. Cada interseção é o encontro de duas vias, cada uma com quatro faixas: duas faixas para tráfego de entrada e duas faixas para tráfego de saída. Há uma faixa para seguir em frente e outra para seguir em frente ou virar à direita, totalizando oito faixas na interseção. As vias são orientadas em sentido duplo.

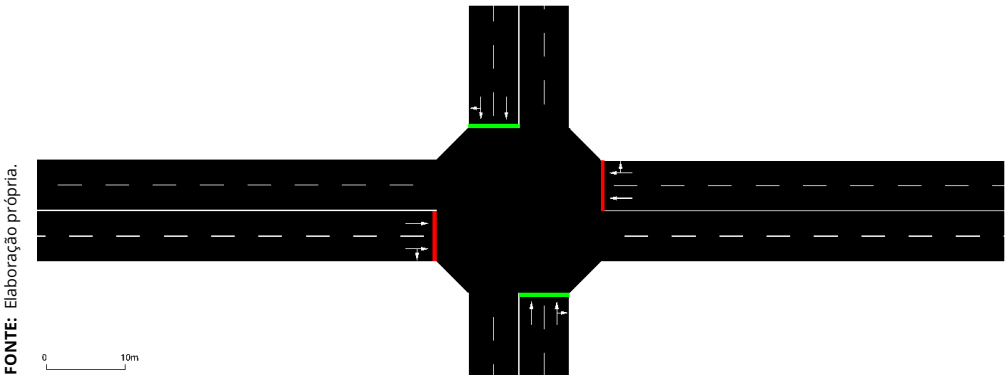


Figura 5.1 – *Interseção simples utilizada na simulação*

5.3.2 Rede com Três Interseções Alinhadas Horizontalmente

A segunda rede de tráfego é mais complexa, composta por três semáforos alinhados horizontalmente ao longo de um corredor principal. Esta configuração visa representar uma seção de uma via arterial urbana. Cada semáforo controla uma interseção, similar à descrita anteriormente, com quatro vias e duas faixas de tráfego em cada direção. As interseções estão espaçadas uniformemente a uma distância de 150 metros entre si, com objetivo de avaliar o impacto que um semáforo tem no fluxo de tráfego nos semáforos subsequentes, causando filas e atrasos em cadeia.

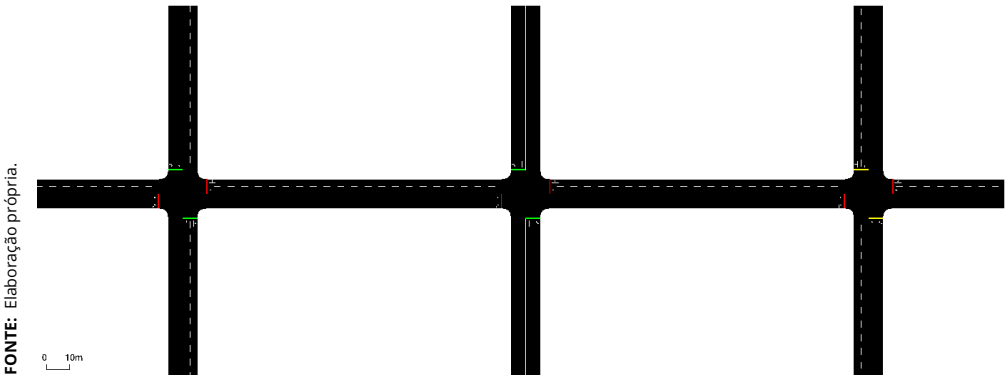


Figura 5.2 – *Rede com três semáforos alinhados horizontalmente utilizada na simulação*

5.4 Semáforos de Tempo Fixo

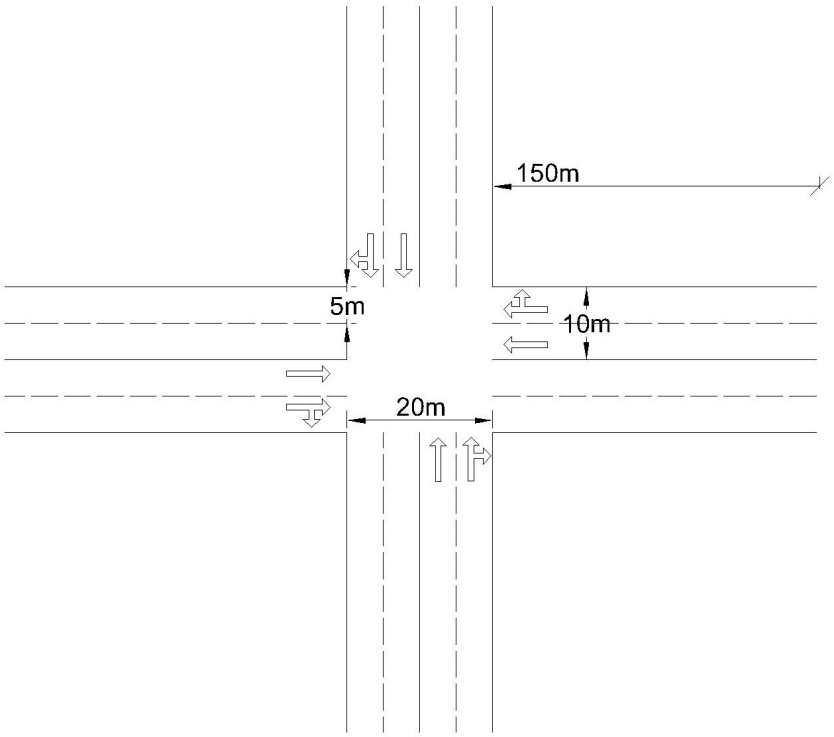
Um semáforo de tempo fixo é um tipo de semáforo de trânsito que opera com ciclos de sinalização pré-determinados e constantes. Isso significa que os tempos

para os sinais verde, amarelo e vermelho são definidos e não mudam com base nas condições de tráfego. No presente tópico, todos os passos descritos foram baseados no Manual Brasileiro de Sinalização de Trânsito – Volume V: Sinalização Semafórica (DENATRAN, 2020).

- **Etapa I**: Definição das condições em que a programação irá operar.
- **Etapa II**: Determinação das características operacionais do tráfego.
- **Etapa III**: Cálculo da programação semafórica.
- **Etapa IV**: Implementação da programação e avaliação dos resultados.

5.4.0.1 Etapa I: Condições de tráfego

a) Levantamento das Características do Local: Para esta simulação, variáveis como geometria, topografia, presença de pedestres e horários de pico não são consideradas. A interseção utilizada é apresentada na Figura 5.3.



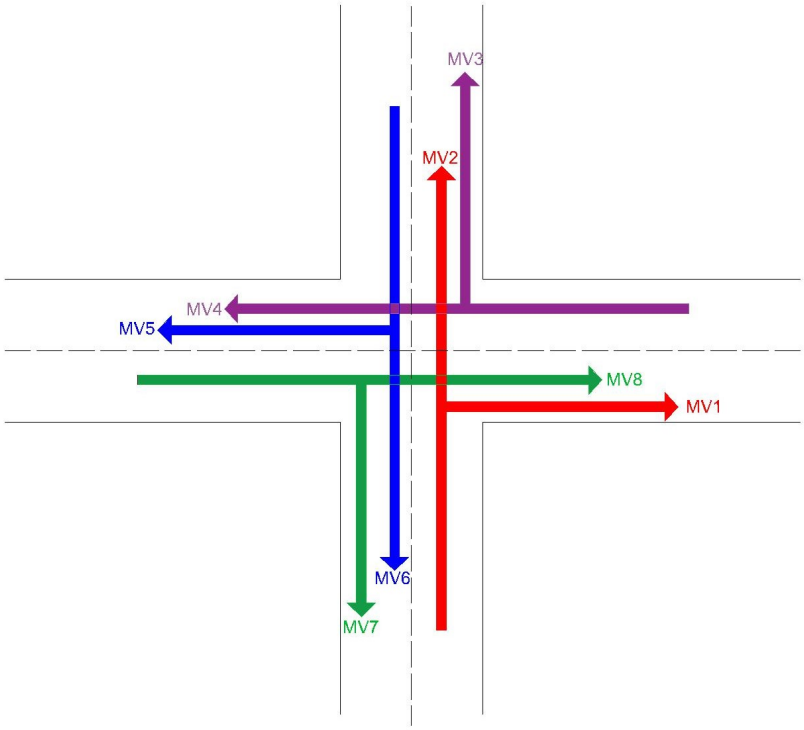
FONTE: Elaboração própria.

Figura 5.3 – *Características da interseção simples utilizada na simulação.*

b) Período de abrangência: Na simulação, os fluxos de veículos foram mantidos constantes durante toda a simulação, sem distinção de períodos específicos.

c) Tempo de ciclo máximo: O tempo de ciclo máximo adotado é de 120 segundos.

d) Análise dos movimentos: Os movimentos considerados (MV) são especificados da seguinte forma: MV1: Sul para Leste, MV2: Sul para Norte, MV3: Leste para Norte, MV4: Leste para Oeste, MV5: Norte para Oeste, MV6: Norte para Sul, MV7: Oeste para Sul, MV8: Oeste para Leste. A representação dos movimentos (MV) é apresentada na Figura 5.4.



FONTE: Elaboração própria.

Figura 5.4 – *Representação dos movimentos.*

Com base nessa análise, foi elaborada a tabela de movimentos conflitantes (Tabela 5.1).

e) Determinação dos grupos de movimentos: Os movimentos de tráfego que compartilham as mesmas vias são agrupados em grupos de movimentos (

f) Definição do diagrama de estágios: O diagrama de estágios é uma representação gráfica que mostra os movimentos agrupados de tráfego permitidos simultaneamente em cada estágio do ciclo semafórico.

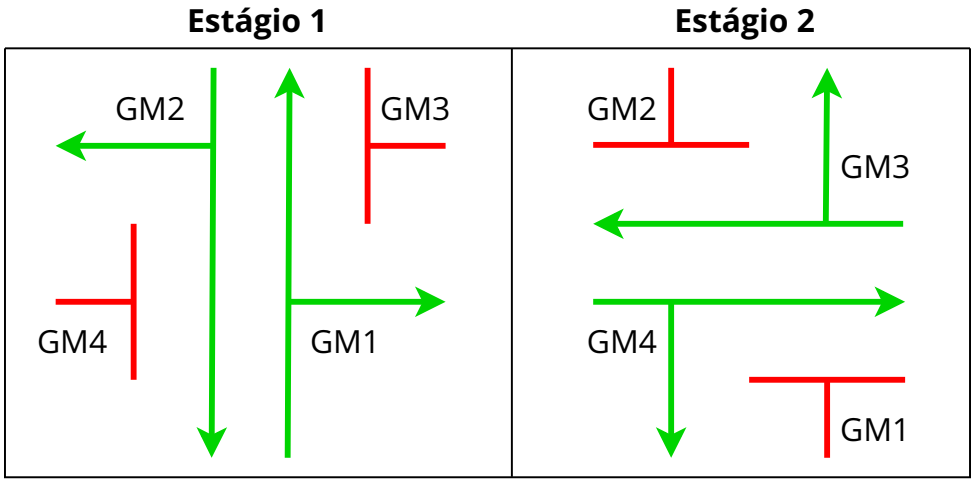
No estágio 1, os grupos GM1 e GM2 estão verdes enquanto GM3 e GM4 estão vermelhos. No estágio 2, os grupos GM3 e GM4 estão verdes enquanto GM1 e GM2 estão vermelhos. A Figura 5.5 ilustra o Diagrama de estágios de um ciclo completo

Tabela 5.1 – *Tabela de Movimentos Conflitantes*

	MV1	MV2	MV3	MV4	MV5	MV6	MV7	MV8
MV1								x
MV2			x	x				x
MV3		x						
MV4		x			x	x		
MV5				x				
MV6				x			x	x
MV7						x		
MV8	x	x				x		

FONTE: Elaboração própria

de operação.



FONTE: Elaboração própria.

Figura 5.5 – *Diagrama de estágios dos semáforos*

g) Definição dos grupos semafóricos: Os grupos semafóricos, conforme definidos no manual, são agrupamentos que atendem os movimentos que recebem as mesmas indicações luminosas ao longo do ciclo.

No simulador SUMO, esses grupos são representados por sequências de caracteres que indicam o estado dos sinais para cada movimento específico. Cada caractere na sequência corresponde a um movimento de tráfego, onde 'G' indica um sinal verde (permissão para avançar) e 'r' indica um sinal vermelho (proibição de avançar).

h) Determinação dos parâmetros de segurança dos grupos de movimentos: O tempo de verde de segurança é de 12 segundos para todos os grupos de movimentos.

Entreverdes: O intervalo entreverdes é um tempo de segurança projetado para prevenir colisões entre veículos que estão terminando seu direito de passagem e aqueles que estão prestes a obtê-lo no próximo estágio. Este período ocorre entre o final do sinal verde de um estágio e o início do sinal verde do estágio seguinte, e pode ser determinado utilizando a Equação 5.1.

$$t_{ent} = t_{pr} + \frac{v}{2(a_{ad} \pm ig)} + \frac{d_2 + c}{v} \tag{5.1}$$

Em que:

- t_{ent} = tempo de entreverdes, em s;
- t_{pr} = tempo de percepção e reação do condutor, em s;
- v = velocidade do veículo, em m/s;
- a_{ad} = máxima taxa de frenagem admissível em via plana ($3,0$ m/s^2);
- i = inclinação da via na aproximação, sendo positivo em rampas ascendentes e negativos em rampas descendentes;
- g = aceleração da gravidade ($9,8$ m/s^2);
- d_2 = extensão da trajetória do veículo entre a linha de retenção e o término da área de conflito, em metros;
- c = comprimento do veículo, em metros.

Ao determinar o tempo de entreverdes, o tempo de vermelho geral, que é o intervalo em que todas as luzes estão vermelhas, pode ser calculado utilizando a última parcela da Equação 5.1, conforme a Equação 5.2:

$$t_{vg} = \frac{d_2 + c}{v} \tag{5.2}$$

Quanto ao tempo de amarelo, este será igual à soma das duas primeiras parcelas da Equação 5.1, conforme a Equação 5.3:

$$t_{am} = t_{pr} + \frac{v}{2(a_{ad} \pm ig)} \tag{5.3}$$

Utilizando os valores sugeridos pelo DENATRAN (DENATRAN, 2020), onde $t_{pr} = 2\,\text{s}$, $v = 13{,}90\,\text{m/s}$ (50 km/h), $d_2 = 15\,\text{m}$, $c = 5\,\text{m}$, e $i = 0$ (superfície plana), e substituindo nas equações de tempo de entreverdes 5.1, tempo de vermelho geral 5.2 e tempo de amarelo 5.3, encontramos os seguintes resultados:

Tempo de vermelho geral:

$$t_{vg} = \frac{d_2 + c}{v} \approx 1\,\text{s} \tag{5.4}$$

Tempo de amarelo:

$$t_{am} = t_{pr} + \frac{v}{2(a_{ad} \pm ig)} \approx 4\,\text{s} \tag{5.5}$$

Tempo de entreverdes:

$$t_{ent} = t_{vg} + t_{am} \approx 5\,\text{s} \tag{5.6}$$

5.4.0.2 Etapa II: Características de tráfego

a) Taxa de Fluxo de Saturação: A taxa de fluxo de saturação (S) é uma medida da capacidade máxima de uma faixa de tráfego em termos de veículos por hora por faixa.

A taxa de fluxo de saturação (FS) é dada por:

$$FS = \frac{3600 \times v}{d} \tag{5.7}$$

A taxa de fluxo de saturação é calculada considerando:

- Velocidade máxima $v = 13{,}9$ m/s
- Comprimento do veículo $c = 5$ m
- Tempo de reação $t_{pr} = 2$ s
- Distância que o veículo percorre desde o momento em que o motorista

identifica uma situação até parar efetivamente. É calculada como:

$$d = c + v \times t_{pr} = 32{,}8 \text{ m} \tag{5.8}$$

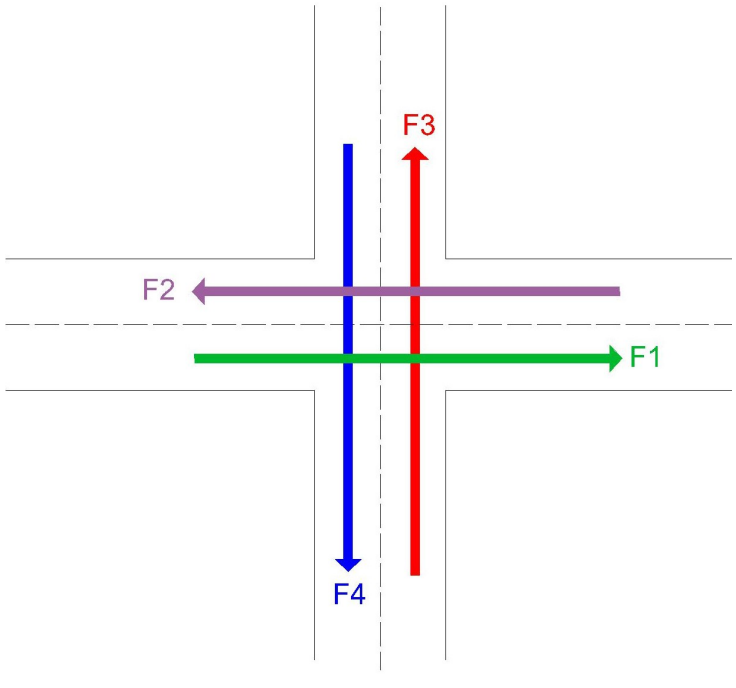
Substituindo os valores apresentados na Equação 5.8 na Equação 5.9, a taxa de fluxo de saturação (FS) é dada por:

$$FS = \frac{3600 \times 13{,}9}{32{,}8} \approx 1525 \text{ veículos/h/faixa} \tag{5.9}$$

Como cada via possui 2 faixas, temos:

$$FS = 2 \times S \approx 3050 \text{ veículos/h/via} \tag{5.10}$$

b) Fluxos de veículos por via: Os fluxos de veículos observados foram capturados utilizando sensores de *loop* no simulador SUMO. Os fluxos que foi definido está exibido na Imagem 5.6.



FONTE: Elaboração própria

Figura 5.6 – *Fluxos de veículos*

O fluxo utilizado foi o maior fluxo observado durante uma simulação de 1 hora. O arquivo que insere os detectores de *loop* no SUMO foi feito conforme o Código 5.1:

Código 5.1 – *Inserção de Detectores de Loop no SUMO*

```
<additional>
  <!-- Detectores para uma via específica -->
  <inductionLoop id="{lane_id}" lane="{lane_id}"
                 pos="{pos}" freq="{freq}"
                 file="{lane_id}.xml"/>
</additional>
```

Os valores de cada fluxo estão listados abaixo:

- $F_1 = 1020$ veículos/h/via
- $F_2 = 1140$ veículos/h/via
- $F_3 = 1080$ veículos/h/via
- $F_4 = 1020$ veículos/h/via

c) Tempo perdido inicial e final de cada grupo de movimentos: Neste exemplo de simulação no SUMO, não houve levantamento de campo dos tempos perdidos devido ao contexto totalmente simulado. Portanto, considerou-se que o tempo perdido de cada grupo de movimentos corresponde ao seu tempo de entreverdes.

5.4.0.3 Etapa III: Cálculo da programação semafórica

a) A taxa de ocupação (y) é uma medida utilizada para avaliar o grau de utilização de uma via ou um sistema de tráfego. Ela é calculada pela Equação 5.11:

$$y = \frac{F}{FS} \tag{5.11}$$

utilizando a Equação 5.11 é possível calcular a taxa de ocupação de cada via:

- $y_1 = \frac{1020}{3050} = 0.33$
- $y_2 = \frac{1140}{3050} = 0.37$
- $y_3 = \frac{1080}{3050} = 0.35$
- $y_4 = \frac{1020}{3050} = 0.33$

b) Definição dos grupos de movimentos críticos: Movimentos críticos são aqueles fluxos de tráfego em uma interseção que têm a maior influência pois são os que apresentam maior demanda ou ocupação.

GM1 e GM2, assim como GM3 e GM4, recebem sinal verde simultaneamente. Portanto, o grupo de movimento crítico deve ser determinado pela maior taxa de ocupação entre os dois grupos de movimentos, sendo y_2 e y_3.

A soma das taxas de ocupação dos grupos críticos é dada pela Equação 5.12.

$$\sum y_i = y_2 + y_3 = 0.37 + 0.35 = 0.72 \tag{5.12}$$

c) Cálculo do Tempo Perdido Total (T_p)

O Tempo Perdido Total (T_p) é o tempo durante o ciclo de sinalização em que nenhum movimento de tráfego está avançando, ou seja, o tempo de amarelo e o tempo de vermelho dos 2 estágios.

Portanto, o tempo perdido total é mostrado na Equação 5.13.

$$T_p = (4 \text{ s de amarelo} + 1 \text{ s de vermelho}) \times 2 = 10 \text{ s} \tag{5.13}$$

d) Cálculo do Tempo de Ciclo e Tempos de Verde: O tempo de ciclo é calculado pelo método do grau de saturação máximo (p_i):

$$p_i = \frac{y_i}{x_{mi}} \tag{5.14}$$

Adotando $x_{m2} = 0.85$ e $x_{m3} = 0.90$, obtêm-se os seguintes resultados:

- $p_2 = \frac{0.37}{0.85} = 0.44$
- $p_3 = \frac{0.35}{0.90} = 0.39$

Somando estes valores, tem-se:

$$\sum p_i = 0.44 + 0.39 = 0.83 \tag{5.15}$$

O tempo de ciclo (t_c), calculado pelo método de saturação máxima, é dado pela Equação 5.16:

$$t_c = \frac{T_p}{1 - \sum p_i} = \frac{10}{1 - 0.83} = 59 \text{ s} \tag{5.16}$$

e) Tempo de Verde Efetivo: O tempo de verde efetivo é calculado por meio da Equação 5.17:

$$t_{\text{verde efetivo}} = p_i \times t_c \tag{5.17}$$

Portanto, os resultados são:

- $t_{\text{verde efetivo para estágio 1}} = 0.44 \times 59 = 26$ s
- $t_{\text{verde efetivo para estágio 2}} = 0.39 \times 59 = 23$ s

5.4.0.4 Etapa IV: Implementação da Programação e Avaliação dos Resultados

Após a realização dos cálculos necessários, foi implementada a programação do semáforo de tempo fixo na plataforma SUMO (Simulation of Urban MObility). A configuração dos tempos de sinalização foi definida utilizando a seguinte lógica de temporização mostrado no Código 5.2:

Código 5.2 – *Lógica de temporização do semáforo*

```
<tlLogic id="t" type="static" programID="0" offset="0">
    <phase duration="23" state="GGGrrrGGGrrr"/>
    <phase duration="4"  state="yyyrrryyyrrr"/>
    <phase duration="1"  state="rrrrrrrrrrrr"/>
    <phase duration="26" state="rrrGGGrrrGGG"/>
    <phase duration="4"  state="rrryyyrrryyy"/>
    <phase duration="1"  state="rrrrrrrrrrrr"/>
</tlLogic>
```

Essa configuração estabelece as seguintes fases de sinalização:

- **Fase 1:** Duração de 23 segundos, com sinal verde para o primeiro conjunto de vias.
- **Fase 2:** Duração de 4 segundos, com sinal amarelo para o primeiro conjunto de vias.
- **Fase 3:** Duração de 1 segundo, com todos os sinais em vermelho.
- **Fase 4:** Duração de 26 segundos, com sinal verde para o segundo conjunto de vias.
- **Fase 5:** Duração de 4 segundos, com sinal amarelo para o segundo conjunto de vias.
- **Fase 6:** Duração de 1 segundo, com todos os sinais em vermelho.

O código foi executado no SUMO para simular o comportamento do tráfego com base nessa lógica de temporização. Durante a simulação, foi observado atentamente o funcionamento do semáforo, verificando-se que operava conforme o esperado. O fluxo de tráfego apresentou um comportamento satisfatório, com tempos de espera dentro dos limites aceitáveis e sem congestionamentos significativos.

5.4.0.5 Coordenação dos Três Semáforos

A coordenação semafórica é a prática de sincronizar múltiplos semáforos ao longo de uma via para otimizar o fluxo de tráfego, reduzindo o tempo de viagem e minimizando paradas desnecessárias. Ela envolve o cálculo de defasagems(atrasos) adequados para cada semáforo, garantindo que os veículos possam se deslocar entre eles de forma contínua e eficiente.

Os tempos de ciclo, verde, amarelo e vermelho foram definidos da mesma forma que na interseção simples apresentada anteriormente. Para os três semáforos, o tempo de amarelo será de 4 segundos e o tempo de vermelho será de 1 segundo. Os demais tempos são mostrados a seguir:

- **Semáforo 1**:

- Tempo de ciclo = 43s
- Tempo verde (Estágio 1) = 11s
- Tempo verde (Estágio 2) = 22s

- **Semáforo 2**:
  - Tempo de ciclo = 43s
  - Tempo verde (Estágio 1) = 11s
  - Tempo verde (Estágio 2) = 22s

- **Semáforo 3**:
  - Tempo de ciclo = 51s
  - Tempo verde (Estágio 1) = 13s
  - Tempo verde (Estágio 2) = 28s

Para determinar a defasagem, primeiro foi calculado o tempo de viagem (T) entre os semáforos. O tempo de viagem é dado pela Equação 5.18:

$$T = \frac{D}{V} \tag{5.18}$$

onde: - D é a distância entre os semáforos, igual a 150 metros. - V é a velocidade média dos veículos, igual a 12,50 m/s.

Substituindo os valores, obtém-se:

$$T = \frac{150\,\text{m}}{12,50\,\text{m/s}} = 12\,\text{s}$$

Em seguida, para calcular a defasagem entre os semáforos, foram considerados os tempos verdes e o tempo de viagem.

O cálculo da defasagem é feito somando o tempo verde do estágio relevante ao tempo de viagem entre os semáforos para garantir a sincronização correta do fluxo de tráfego.

A defasagem entre o Semáforo 1 (t1) e o Semáforo 2 (t2) é dada por:

$$\text{Defasagem}_{12} = 22s + 12s = 34s$$

Analogamente, a defasagem entre o Semáforo 2 (t2) e o Semáforo 3 (t3) é:

$$\text{Defasagem}_{23} = 28s + 12s = 40s$$

As defasagens (*offset*) calculadas foram implementadas nos controladores de semáforo no Pseudocódigo 5.3:

Código 5.3 – *Lógica final de temporização dos semáforos*

```
<tlLogic id="t1" type="static" programID="0" offset="0">
    <phase duration="11" state="GGGrrrGGGrrr"/>
    <phase duration="4"  state="yyyrrryyyrrr"/>
    <phase duration="1"  state="rrrrrrrrrrrr"/>
    <phase duration="22" state="rrrGGGrrrGGG"/>
    <phase duration="4"  state="rrryyyrrryyy"/>
    <phase duration="1"  state="rrrrrrrrrrrr"/>
</tlLogic>
<tlLogic id="t2" type="static" programID="0" offset="34">
    <phase duration="11" state="GGGrrrGGGrrr"/>
    <phase duration="4"  state="yyyrrryyyrrr"/>
    <phase duration="1"  state="rrrrrrrrrrrr"/>
    <phase duration="22" state="rrrGGGrrrGGG"/>
    <phase duration="4"  state="rrryyyrrryyy"/>
    <phase duration="1"  state="rrrrrrrrrrrr"/>
</tlLogic>
<tlLogic id="t3" type="static" programID="0" offset="40">
    <phase duration="13" state="GGGrrrGGGrrr"/>
    <phase duration="4"  state="yyyrrryyyrrr"/>
    <phase duration="1"  state="rrrrrrrrrrrr"/>
    <phase duration="28" state="rrrGGGrrrGGG"/>
    <phase duration="4"  state="rrryyyrrryyy"/>
    <phase duration="1"  state="rrrrrrrrrrrr"/>
</tlLogic>
```

A Figura 5.7 mostra a representação do diagrama de tempo e espaço resultante. A banda representa o intervalo em que os veículos podem passar sem encontrar sinal vermelho, que foi de 4 segundos."

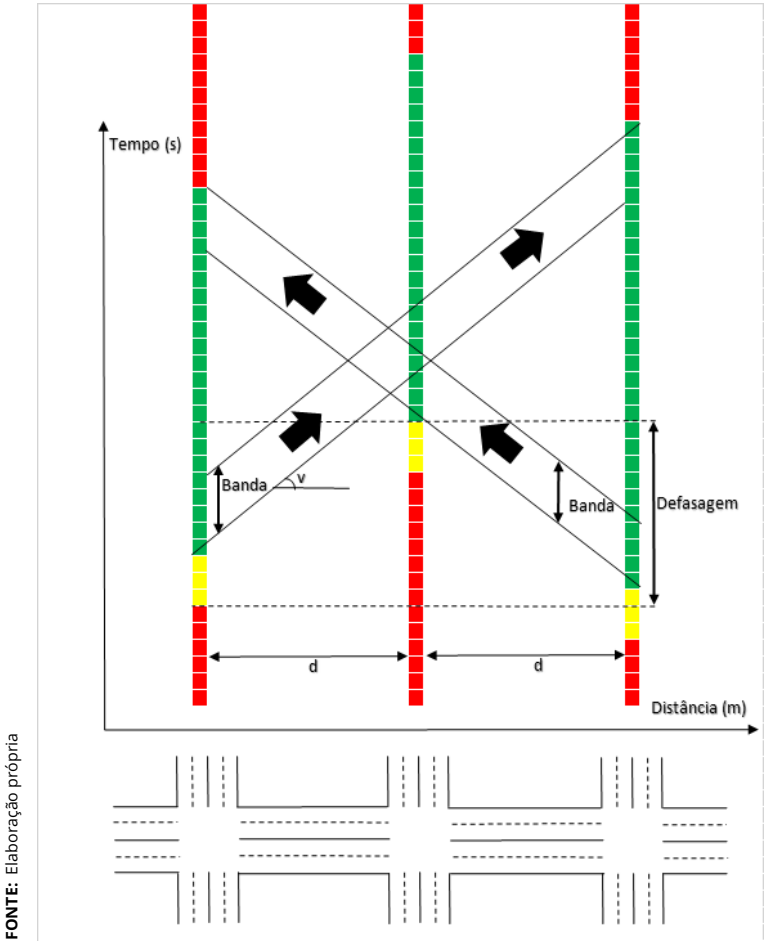


Figura 5.7 – *Diagrama espaço-tempo para as vias de mão dupla*

Dessa forma, assegurou-se que os tempos de viagem fossem considerados, permitindo que os veículos passassem pelos semáforos de forma contínua, minimizando paradas.

5.5 Semáforos de Aprendizagem por Reforço

5.5.1 Funções de Recompensa

No sistema de controle de tráfego utilizando aprendizado por reforço, a recompensa é calculada para reduzir os tempos de espera dos veículos. A função de recompensa é especificada como um dicionário, onde cada semáforo tem uma

função individual, permitindo uma personalização precisa para cada cruzamento.

Cada sinal de tráfego é inicializado com a função de recompensa correspondente do dicionário, garantindo que cada semáforo utilize a lógica apropriada para sua situação.

As recompensas são calculadas utilizando a função *'getAccumulatedWaitingTime'* do TRACI, que calcula o tempo de espera acumulado dos veículos em cada semáforo. Isso assegura que as recompensas sejam atualizadas de maneira dinâmica, refletindo o desempenho atual do sistema de tráfego.

A Equação 5.19 foi utilizada para calcular a recompensa é baseada na diferença no tempo de espera dos veículos:

$$R_t = W_{t-1} - W_t \tag{5.19}$$

Onde:

- R_t é a recompensa no tempo t.
- W_t é o tempo de espera total dos veículos no tempo t.
- W_{t-1} é o tempo de espera total dos veículos no tempo $t-1$.

5.5.2 Função de Observação

A observação foi configurada para obter o estado atual do tráfego, permitindo que o semáforo tomasse decisões com base no tráfego atual. A função de observação foi codificada como um dicionário, onde cada semáforo teve sua própria função de observação. Cada sinal de tráfego foi inicializado com a função de observação correspondente especificada no dicionário.

$s =$ [fase atual, densidade de cada faixa, fila de cada faixa, velocidade dos veículos]

Para a observação foram utilizados os seguintes componentes:

- Fase atual do semáforo.

- Densidade de cada faixa, obtida pela função `'getLanesDensity'`.
- Fila de cada faixa, obtida pela função `'getLanesQueue'`.
- Velocidade de cada veículo, obtida pela função `'getSpeed'`

5.6 Definição dos Parâmetros α, γ, ϵ

A definição dos parâmetros α, γ e ϵ foi realizada por meio de ajustes iterativos baseados em valores extraídos da literatura, como os propostos por (SUTTON; BARTO, 2018). O ajuste desses parâmetros é essencial para otimizar o comportamento do agente e garantir um aprendizado eficiente.

Inicialmente, a taxa de aprendizado α foi definida como 0,1, um valor que se mostrou adequado após testes iniciais, baseados em trabalhos como (MNIH et al., 2015b), que sugerem que uma taxa de aprendizado moderada promove a estabilidade do algoritmo sem comprometer a capacidade de adaptação do agente. Durante os testes, valores mais altos levaram a oscilações no aprendizado, tornando o ajuste necessário para evitar a instabilidade.

O fator de desconto γ foi ajustado para 0,95, com base em simulações que indicaram que valores inferiores prejudicavam o planejamento de longo prazo. Estudos como o de (WATKINS; DAYAN, 1992) também apontam que um fator de desconto elevado é necessário para que o agente possa otimizar suas decisões ao longo do tempo, especialmente em ambientes dinâmicos. Nos testes, valores superiores a 0,95 não apresentaram ganhos significativos, mas retardavam a convergência, o que levou à escolha desse valor como ideal.

O parâmetro ϵ, que controla a exploração, foi configurado inicialmente em 1,0, seguindo uma abordagem inspirada no método ϵ-greedy. Durante o treinamento, observou-se que a manutenção de uma alta taxa de exploração por muito tempo resultava em escolhas aleatórias excessivas. A solução encontrada foi um decaimento gradual, com uma taxa de 0,99, levando o ϵ a um valor final de 0,1. Esse ajuste permitiu que o agente explorasse o ambiente de forma controlada, mas ainda aproveitasse o conhecimento adquirido, conforme recomendado por (TOKIC, 2010).

A Tabela 5.2 apresenta os valores finais adotados para os parâmetros após o processo de ajuste:

Tabela 5.2 – *Parâmetros utilizados no algoritmo*

Parâmetro	Valor
α	0.1
γ	0.95
$\epsilon_{\text{inicial}}$	1.0
ϵ_{final}	0.1
Taxa de decaimento	0.99

FONTE: Elaboração própria.

5.7 Modelo Q-learning

O treinamento com Q-Learning para controle de semáforos foi realizado por meio de um processo estruturado, permitindo que os agentes aprendessem a otimizar o fluxo de tráfego. O treinamento consistiu nas seguintes etapas principais:

Primeiramente, o ambiente de simulação foi reinicializado no início de cada execução para garantir que as condições iniciais fossem consistentes. Os estados iniciais dos semáforos foram obtidos e codificados para serem utilizados pelos agentes.

Código 5.4 – *Reset do ambiente de simulação*

```
Para cada execucao:
Reinicializar o ambiente de simulacao
Obter e codificar os estados iniciais dos semaforos
```

Durante o treinamento, os agentes tomaram ações com base no estado atual do ambiente. Essas ações consistiram em decidir a duração dos sinais verdes para cada direção do tráfego no cruzamento. Após cada ação, o ambiente retornou um novo estado e uma recompensa, que foram utilizados para atualizar as tabelas Q dos agentes. Esse processo foi repetido até que a simulação fosse concluída, permitindo que os agentes ajustassem continuamente suas políticas de controle.

Código 5.5 – *Execução dos passos de treinamento*

```
Enquanto nao terminado:
Para cada agente:
Selecionar acao com base na estrategia epsilon-greedy
Executar acao no ambiente
Obter novo estado e recompensa
```

```
6 Atualizar tabela Q
```

Os agentes utilizaram o método de *Q-Learning* para atualizar suas tabelas Q. Esta atualização foi baseada na recompensa recebida e no novo estado observado. A fórmula do *Q-Learning* foi aplicada para ajustar os valores na tabela Q, promovendo o aprendizado incremental dos agentes ao longo do treinamento.

Código 5.6 – *Aprendizado dos agentes Q-Learning*

```
1 Para cada novo estado e recompensa recebida:
2 Atualizar valor Q usando a fórmula Q-Learning:
3 Q(s, a) = Q(s, a) + alpha * (reward + gamma * max(Q(s', a')) - Q(s, a))
4 Atualizar o estado atual do agente para o novo estado
```

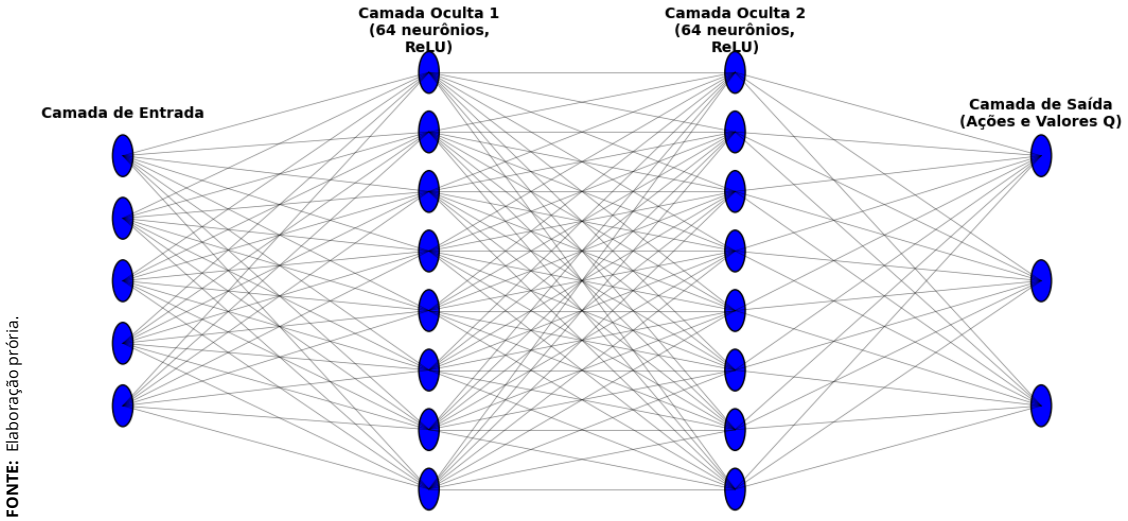
Este treinamento foi repetido para cada execução configurada, permitindo que os agentes de *Q-Learning* aprendessem e otimizassem o controle dos semáforos ao longo do tempo. Ao final de cada execução, os resultados foram salvos para análise posterior, permitindo uma avaliação detalhada do desempenho dos agentes e a identificação de áreas para melhoria.

5.8 Modelo *Deep Q-Network*

A rede neural utilizada no modelo *Deep Q-Network* (DQN) é configurada com a política `MlpPolicy`, fornecida pela biblioteca *Stable Baselines3*. A arquitetura desta rede neural é composta por uma camada de entrada, duas camadas ocultas com 64 neurônios cada, e uma camada de saída. A seguir, são detalhadas as funções e características de cada camada.

A Figura 5.8 ilustra a arquitetura detalhada da rede neural `MlpPolicy` utilizada no modelo DQN. A imagem mostra as camadas da rede neural, incluindo as duas camadas ocultas, cada uma com 64 neurônios e a função de ativação ReLU, conectadas sequencialmente da camada de entrada à camada de saída.

A **camada de entrada** é responsável por receber os dados de estado do ambiente de tráfego. Esta camada não realiza processamento complexo, apenas repassa as informações recebidas para a primeira camada oculta. Os neurônios na camada de entrada correspondem diretamente às características de entrada do



FONTE: Elaboração própria.

Figura 5.8 – *Arquitetura da Rede Neural MlpPolicy Utilizada no Modelo DQN*

ambiente, como configurações iniciais, comprimento da fila, tempo de espera, os tempos dos semáforos e tempo desde a ultima mudança.

A **primeira camada oculta** realiza o primeiro nível de processamento e transformação dos dados recebidos da camada de entrada. Esta camada é composta por 64 neurônios e utiliza a função de ativação ReLU. Esta função de ativação introduz não-linearidades no modelo, permitindo que a rede aprenda representações mais complexas dos dados de entrada.

A **segunda camada oculta** realiza processamento adicional sobre os dados transformados pela primeira camada oculta. Similarmente à primeira camada oculta, esta camada também possui 64 neurônios e utiliza a função de ativação ReLU. A adição de uma segunda camada oculta permite que a rede neural capture padrões e dependências ainda mais complexos nos dados de entrada.

A **camada de saída** é responsável por produzir a decisão final da rede neural. No contexto do DQN, esta camada estima os valores Q para cada ação possível, dada a representação interna do estado do tráfego fornecida pelas camadas ocultas. Cada neurônio na camada de saída corresponde a uma ação possível, como mudar um semáforo para verde, vermelho, ou manter o estado atual. Os valores Q produzidos indicam a expectativa de recompensa futura para cada ação, guiando a escolha da ação mais apropriada.

O modelo DQN aprende ao longo de *timesteps* definidos, controlando os passos da simulação e atualizando os estados com base no aprendizado obtido. No código, isso é realizado da seguinte forma:

A simulação é executada em passos discretos, onde a cada passo, os estados são atualizados com base nas ações do modelo DQN e nas observações do ambiente, conforme Pseudocódigo 5.7.

Código 5.7 – *Controle de Passos*

```
Para cada passo da simulacao:
    Observar estado atual
    Selecionar acao usando epsilon-greedy
    Executar acao
    Observar novo estado e recompensa
    Armazenar transicao no buffer de replay
    Amostrar lote do buffer de replay
    Atualizar rede neural Q
```

Durante cada passo da simulação, o modelo DQN observa o estado atual, seleciona uma ação com base na política *epsilon-greedy*, executa a ação e observa o novo estado e a recompensa resultante. As transições são armazenadas em um *buffer* de *replay*, permitindo que o modelo aprenda a partir de um conjunto de experiências diversificadas. O uso do *buffer* de *replay* é essencial para quebrar a correlação entre experiências sequenciais, promovendo um aprendizado mais robusto.

A política *epsilon-greedy* é aplicada para garantir que o agente explore suficientemente o espaço de estados. No início do treinamento, o valor de epsilon é alto, permitindo que o agente explore diversas ações. À medida que o treinamento avança, epsilon é gradualmente reduzido, incentivando o agente a explorar mais as ações que já se mostraram eficazes.

Atualização da Rede Neural Q: A rede neural Q é atualizada usando um lote de amostras retiradas do *buffer* de *replay*. Este processo ajuda a estabilizar o aprendizado e a evitar correlações indesejadas entre as amostras de treinamento. A equação de Bellman, aplicada aqui, é utilizada para ajustar os pesos da rede neural com base nas recompensas recebidas e nas previsões de recompensas futuras cujo código é mostrado no Pseudocódigo 5.8.

Código 5.8 – *Equação de Bellman*

```
Para cada passo da simulação:
Q(s, a)=Q(s, a) + alpha*(reward + gamma * max(Q(s', a')) - Q(s, a))
```

A atualização da rede neural Q é um processo iterativo onde os pesos da rede são ajustados para minimizar a diferença entre a recompensa prevista e a recompensa observada. Este ajuste contínuo permite que o modelo DQN melhore suas previsões e, consequentemente, suas decisões de controle de tráfego ao longo do tempo.

5.9 Treinamento dos Modelos

O treinamento dos modelos *Q-Learning* e DQN foi conduzido até que os algoritmos atingissem uma política estável. A estabilização foi identificada quando as variações nas recompensas entre os episódios, indicando que o modelo não apresentava mais melhorias relevantes.

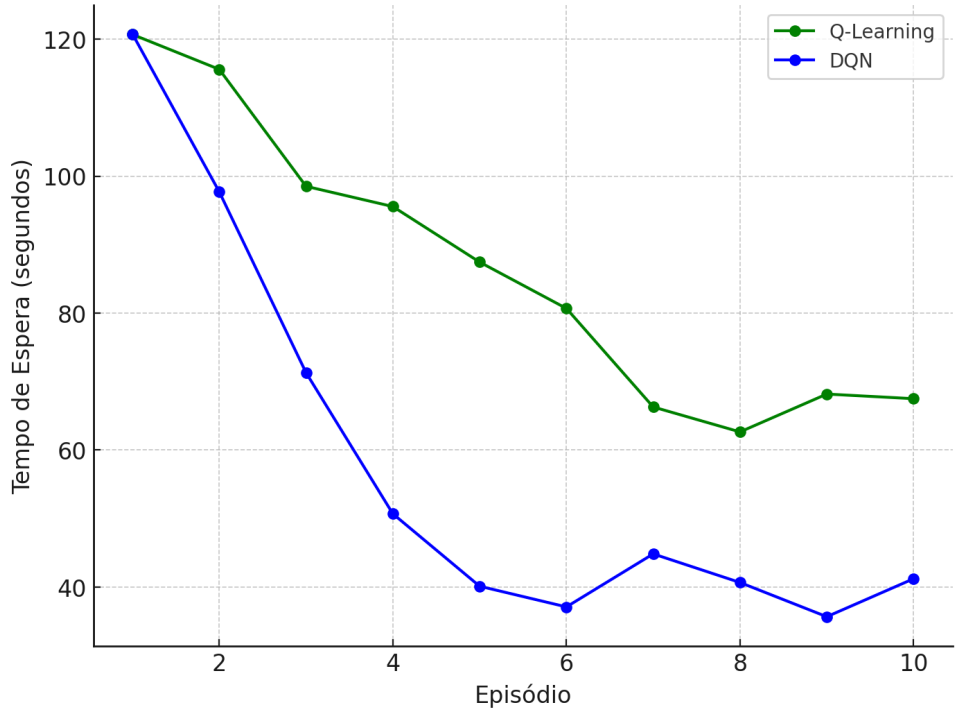
5.9.1 Interseção Simples

Ambos os algoritmos foram configurados inicialmente com 10 episódios e 100.000 *steps* por episódio. A estabilização ocorreu em momentos diferentes para cada modelo, observada pela consistência no comportamento do agente e pela ausência de variações significativas no desempenho.

Devido à baixa complexidade da interseção simples, foram configurados 100.000 *steps* por episódio e 10 episódios. A simplicidade do ambiente permitiu que episódios mais longos fossem suficientes para explorar o espaço de estados de forma eficaz, necessitando de menos repetições.

- ***Q-Learning*:** Estabilizou-se no episódio 7, após o qual o tempo de espera dos veículos e as recompensas não apresentaram mudanças significativas. O tempo de treinamento foi estimado entre 10 a 15 horas.
- **DQN:** O modelo convergiu mais rapidamente, estabilizando-se no episódio 4, devido ao uso da rede neural, que capturou o padrão do ambiente mais cedo. O tempo de treinamento foi estimado entre 20 a 30 horas.

Os dois algoritmos de aprendizado por reforço, Q-Learning e DQN, começaram com tempos de espera elevados no início dos episódios. À medida que o treinamento progrediu, ambos os algoritmos apresentaram melhorias consistentes na redução do tempo de espera dos veículos. No entanto, o DQN apresentou uma convergência mais rápida em comparação com o Q-Learning. O Gráfico 5.9 ilustra claramente a evolução do tempo de espera ao longo dos episódios, destacando o comportamento de cada algoritmo durante o processo de aprendizado.



FONTE: Elaboração própria

Figura 5.9 – *Treinamento dos modelos de aprendizagem por reforço na interseção simples*

5.9.2 3 Interseções Alinhadas

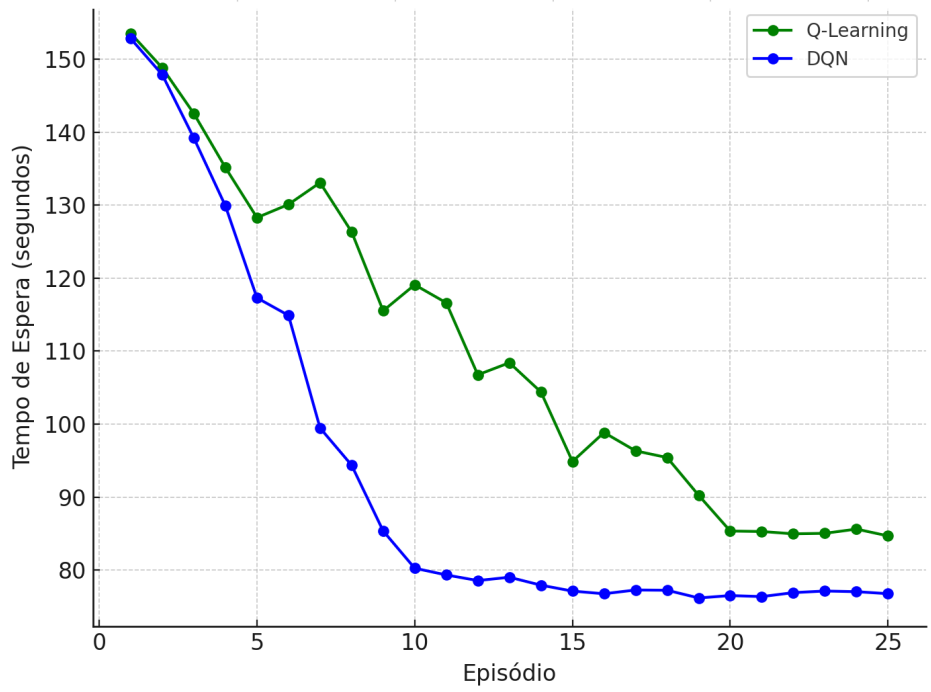
Neste ambiente mais complexo, foram configurados 25 episódios e 30.000 *steps* por episódio. A estabilização foi observada com base na consistência das recompensas e na fluidez do tráfego, indicando que a política ótima foi alcançada.

Para as 3 interseções alinhadas, um ambiente mais complexo, foram utilizados 30.000 *steps* por episódio com 25 episódios. A complexidade exigiu mais repetições para ajustar as políticas devido às interações entre os múltiplos semáforos.

- ***Q-Learning*:** Estabilizou-se no episódio 19, refletindo a necessidade de mais episódios devido à maior complexidade das interseções. O tempo de treinamento foi estimado entre 20 a 30 horas.
- **DQN:** A convergência foi atingida no episódio 11, com a rede neural aprendendo a política mais rapidamente. O tempo de treinamento foi estimado entre 40 a 60 horas, devido ao maior custo computacional.

Assim como no cenário de interseção simples, o Gráfico 5.10 mostram que,

no ambiente de 3 interseções alinhadas, ambos os algoritmos de aprendizado por reforço começaram com tempos de espera elevados nos primeiros episódios. Com o progresso do treinamento, houve uma melhora significativa na redução desses tempos, conforme os algoritmos aprenderam a otimizar o controle dos semáforos. O DQN, mais uma vez, demonstrou uma convergência mais rápida em relação ao *Q-Learning*, estabilizando-se no episódio 11, enquanto o *Q-Learning* continuou a apresentar melhorias até o episódio 19. O Gráfico 5.10 apresenta a evolução dos tempos de espera ao longo dos episódios, evidenciando o desempenho de cada algoritmo neste ambiente mais complexo.



FONTE: Elaboração própria

Figura 5.10 – *Treinamento dos modelos de aprendizagem por reforço no cenário de 3 semáforos*

Capítulo 6

Resultados e discussões

Nesta seção, apresentamos a análise dos resultados finais após o treinamento dos sistemas de controle de tráfego tanto em interseção simples como em três interseções alinhadas horizontalmente. Os dados serão analisados com base em três métricas principais: total de paradas, tempo total de espera e velocidade média do sistema. Para cada métrica, apresentamos uma tabela com as médias e desvios padrão, seguida de um gráfico que compara os três sistemas (Tempo Fixo, *Q-learning* e DQN). As linhas ajustadas nos gráficos indicam a tendência geral dos dados ao longo do tempo, permitindo uma comparação visual entre os diferentes métodos. Os gráficos foram plotados juntos para facilitar essa comparação.

6.1 Interseção Simples

6.1.1 Total de Paradas

Tabela 6.1 – *Médias e Desvios Padrão para Total de Paradas*

Sistema	Média	Desvio Padrão
Tempo Fixo	9.33	4.72
Q-learning	8.19	3.90
DQN	7.01	3.77

A Tabela 6.1 mostra as médias e desvios padrão para o total de paradas. Após o treinamento, o sistema de controle de tráfego baseado em DQN apresentou a menor média de paradas totais, com 7.01 paradas, indicando que o sistema con-

seguiu aprender a minimizar o número de vezes que os veículos precisam parar. Comparativamente, o *Q-learning* teve uma média de 8.19 paradas, mostrando também uma redução significativa em relação ao Tempo Fixo, que apresentou uma média de 9.33 paradas.

O sistema DQN reduziu o número de paradas em 24.86% em comparação com o Tempo Fixo e em 14.39% em comparação com o *Q-learning*. Já o *Q-learning* apresentou uma redução de 12.21% em relação ao Tempo Fixo.

O desvio padrão das paradas é relativamente próximo entre os três sistemas, sendo ligeiramente maior no Tempo Fixo. Isso sugere que, enquanto o *Q-learning* e o DQN conseguiram reduzir as paradas, a variabilidade em seu desempenho é maior no Tempo Fixo, possivelmente devido à sua incapacidade de se adaptar às mudanças nas condições de tráfego.

A redução no número de paradas totais implica diretamente em um fluxo de tráfego mais eficiente, reduzindo o tempo de viagem e melhorando a experiência dos motoristas. Sistemas que minimizam as paradas também contribuem para a diminuição do consumo de combustível e emissões de poluentes, pois menos paradas e arranques resultam em menor gasto de combustível. Portanto, a implementação de sistemas de controle de tráfego baseados em DQN pode levar a uma operação de tráfego mais suave e sustentável.

A Figura 6.1 mostra o gráfico do total de paradas versus o número de *steps* para os três sistemas.

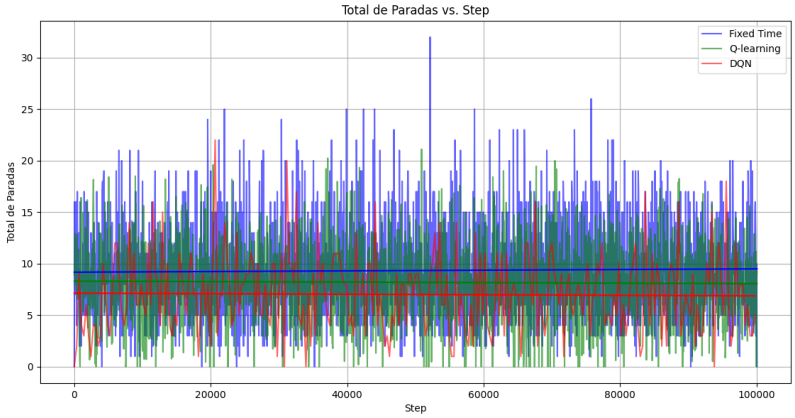


Figura 6.1 – *Total de Paradas vs. Step*

6.1.2 Tempo Total de Espera

Tabela 6.2 – *Médias e Desvios Padrão para Tempo Total de Espera*

Sistema	Média	Desvio Padrão
Tempo Fixo	95.29	75.62
Q-learning	66.61	33.12
DQN	39.24	27.89

A Tabela 6.2 mostra as médias e desvios padrão para o tempo total de espera. O tempo total de espera é um indicador da eficiência de um sistema de controle de tráfego. O sistema DQN, com uma média de 39.24 segundos de espera, demonstra a melhor performance, seguido pelo *Q-learning* com uma média de 66.61 segundos, e o Tempo Fixo com uma média de 95.29 segundos.

O sistema DQN reduziu o tempo de espera em 58.82% em comparação com o Tempo Fixo e em 41.10% em comparação com o *Q-learning.* Já o *Q-learning* apresentou uma redução de 30.09% em relação ao Tempo Fixo.

A menor variabilidade (desvio padrão) no tempo de espera do DQN sugere um desempenho mais consistente, com menos flutuações. Em comparação, o Tempo Fixo apresenta a maior variabilidade, indicando inconsistências no tempo de espera dos veículos.

Menores tempos de espera são benéficos para os usuários das vias, reduzindo o tempo total de viagem e a frustração associada a longos períodos de espera nos semáforos. Além disso, um menor tempo de espera contribui para a eficiência energética dos veículos e a redução das emissões de gases poluentes. A adoção de sistemas DQN pode, portanto, melhorar a qualidade de vida urbana ao otimizar o fluxo de tráfego e reduzir o impacto ambiental.

A Figura 6.2 mostra o gráfico do tempo total de espera versus o número de *steps* para os três sistemas.

6.1.3 Velocidade Média do Sistema

A Tabela 6.3 mostra as médias e desvios padrão para a velocidade média do sistema. A velocidade média do sistema é uma métrica importante, mas deve ser analisada em conjunto com outras métricas como paradas e tempos de espera. O

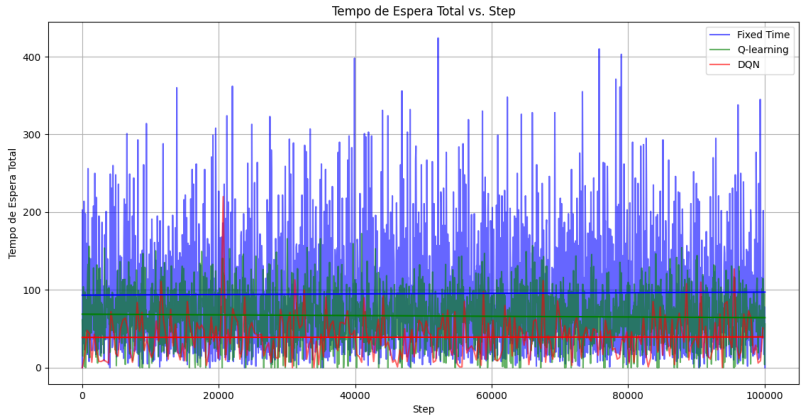


Figura 6.2 – *Tempo de Espera Total vs. Step*

Tabela 6.3 – *Médias e Desvios Padrão para Velocidade Média*

Sistema	Média	Desvio Padrão
Tempo Fixo	7.88	1.63
Q-learning	7.99	0.77
DQN	7.89	1.37

sistema *Q-learning* apresentou a maior velocidade média (7.99), seguido pelo DQN (7.89), e pelo Tempo Fixo (7.88).

O sistema *Q-learning* aumentou a velocidade média em 1.40% em comparação com o Tempo Fixo e em 1.27% em comparação com o DQN. Já o DQN apresentou um aumento de 0.13% em relação ao Tempo Fixo.

O desvio padrão da velocidade média é maior no Tempo Fixo, indicando uma maior variabilidade nas velocidades dos veículos. Essa variabilidade pode resultar de ajustes dinâmicos e estratégias de aprendizado contínuo do *Q-learning*, que tenta melhorar seu desempenho ao longo do tempo e adaptar-se às mudanças nas condições de tráfego.

Embora uma maior velocidade média possa parecer positiva, é importante considerar que um sistema que otimiza apenas para velocidade pode resultar em maiores paradas e tempos de espera, prejudicando a eficiência geral. O equilíbrio encontrado nos sistemas DQN e *Q-learning*, que minimizam paradas e tempos de espera, sugere uma abordagem mais eficaz para o controle de tráfego. A velocidade média mais baixa do Tempo Fixo é compensada por melhorias em outras métricas, resultando em um sistema de tráfego mais equilibrado e eficiente.

A Figura 6.3 mostra o gráfico da velocidade média versus o número de *steps* para os três sistemas.

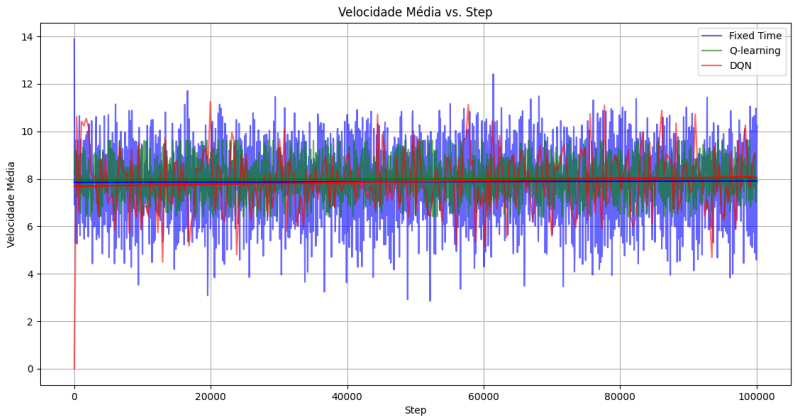


Figura 6.3 – *Velocidade Média vs. Step*

6.2 Três Interseções Alinhados Horizontalmente

6.2.1 Total de Paradas

A Tabela 6.4 mostra as médias e desvios padrão para o total de paradas. Após o treinamento, o sistema de controle de tráfego baseado em DQN apresentou a menor média de paradas totais, com 9.74 paradas, indicando que o sistema conseguiu aprender a minimizar o número de vezes que os veículos precisam parar. Comparativamente, o *Q-learning* teve uma média de 10.96 paradas, mostrando também uma redução significativa em relação ao Tempo fixo, que apresentou uma média de 12.20 paradas.

Tabela 6.4 – *Médias e Desvios Padrão para Total de Paradas*

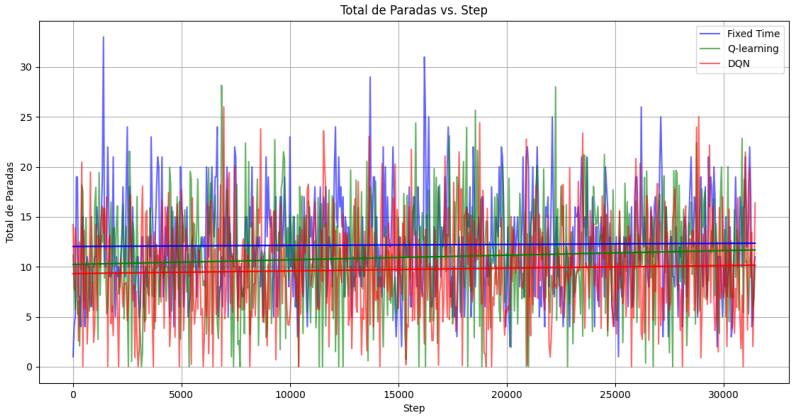
Sistema	Média	Desvio Padrão
Tempo fixo	12.20	4.90
Q-learning	10.96	5.26
DQN	9.74	5.08

O sistema DQN reduziu o número de paradas em 20.16% em comparação com o Tempo Fixo e em 11.12% em comparação com o *Q-learning*. Já o *Q-learning* apresentou uma redução de 10.16% em relação ao Tempo Fixo.

O desvio padrão das paradas é relativamente próximo entre os três sistemas, sendo ligeiramente maior no *Q-learning*. Isso sugere que, enquanto o *Q-learning* conseguiu reduzir as paradas, a variabilidade em seu desempenho é maior, possivelmente devido a flutuações em condições de tráfego diferentes durante o treinamento e adaptação às mudanças nas condições de tráfego ao longo do tempo.

A redução no número de paradas totais implica diretamente em um fluxo de tráfego mais eficiente, reduzindo o tempo de viagem e melhorando a experiência dos motoristas. Sistemas que minimizam as paradas também contribuem para a diminuição do consumo de combustível e emissões de poluentes, pois menos paradas e arranques resultam em menor gasto de combustível. Portanto, a implementação de sistemas de controle de tráfego baseados em DQN pode levar a uma operação de tráfego mais suave e sustentável.

A Figura 6.4 mostra o gráfico do total de paradas versus o número de *steps* para os três sistemas.



FONTE: Elaboração própria

Figura 6.4 – *Total de Paradas vs. Step*

6.2.2 Tempo Total de Espera

A Tabela 6.5 mostra as médias e desvios padrão para o tempo total de espera. O tempo total de espera é um indicador da eficiência de um sistema de controle de tráfego. O sistema DQN, com uma média de 76.94 segundos de espera, demonstra a melhor performance, seguido pelo *Q-learning* com uma média de 84.97 segundos, e o Tempo fixo com uma média de 94.94 segundos.

O sistema DQN reduziu o tempo de espera em 18.98% em comparação com

Tabela 6.5 – *Médias e Desvios Padrão para Tempo Total de Espera*

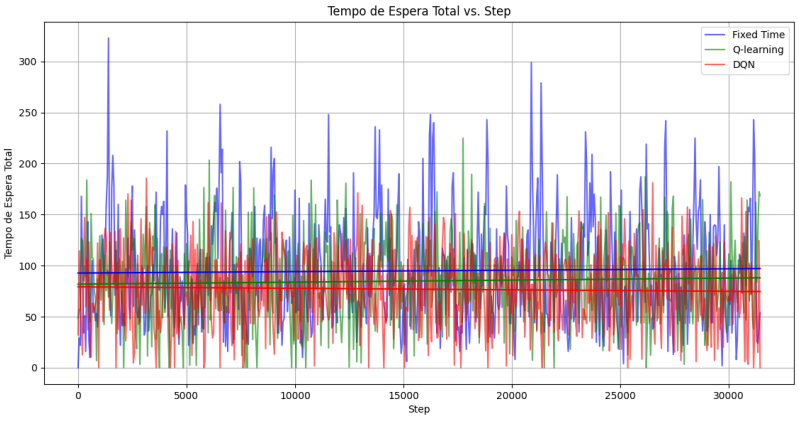
Sistema	Média	Desvio Padrão
Tempo fixo	94.94	53.61
Q-learning	84.97	41.44
DQN	76.94	39.00

o Tempo Fixo e em 9.45% em comparação com o *Q-learning*. Já o *Q-learning* apresentou uma redução de 10.49% em relação ao Tempo Fixo.

A menor variabilidade (desvio padrão) no tempo de espera do DQN sugere um desempenho mais consistente, com menos flutuações. Em comparação, o Tempo fixo apresenta a maior variabilidade, indicando inconsistências no tempo de espera dos veículos.

Menores tempos de espera são benéficos para os usuários das vias, reduzindo o tempo total de viagem e a frustração associada a longos períodos de espera nos semáforos. Além disso, um menor tempo de espera contribui para a eficiência energética dos veículos e a redução das emissões de gases poluentes. A adoção de sistemas DQN pode, portanto, melhorar a qualidade de vida urbana ao otimizar o fluxo de tráfego e reduzir o impacto ambiental.

A Figura 6.5 mostra o gráfico do tempo total de espera versus o número de *steps* para os três sistemas



FONTE: Elaboração própria

Figura 6.5 – *Tempo de Espera Total vs. Step*

6.2.3 Velocidade Média do Sistema

A Tabela 6.6 mostra as médias e desvios padrão para a velocidade média do sistema. A velocidade média do sistema é uma métrica importante, mas deve ser analisada em conjunto com outras métricas como paradas e tempos de espera. O sistema Tempo fixo apresentou a maior velocidade média (7.32), seguido pelo *Q-learning* (6.93), e pelo DQN (5.97).

Tabela 6.6 – *Médias e Desvios Padrão para Velocidade Média*

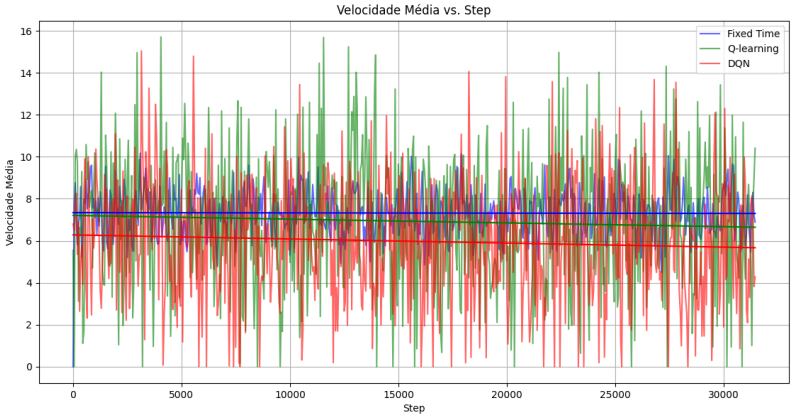
Sistema	Média	Desvio Padrão
Tempo fixo	7.32	1.09
Q-learning	6.93	3.17
DQN	5.97	2.92

O sistema Tempo Fixo aumentou a velocidade média em 5.62% em comparação com o *Q-learning* e em 22.61% em comparação com o DQN. Já o *Q-learning* apresentou um aumento de 16.09% em relação ao DQN.

O desvio padrão da velocidade média é maior nos sistemas de aprendizado por reforço, particularmente no *Q-learning*, indicando uma maior variabilidade nas velocidades dos veículos. Essa variabilidade pode resultar de ajustes dinâmicos e estratégias de aprendizado contínuo do *Q-learning*, que tenta melhorar seu desempenho ao longo do tempo e adaptar-se às mudanças nas condições de tráfego.

Embora uma maior velocidade média possa parecer positiva, é importante considerar que um sistema que otimiza apenas para velocidade pode resultar em maiores paradas e tempos de espera, prejudicando a eficiência geral. O equilíbrio encontrado nos sistemas DQN, que minimizam paradas e tempos de espera, sugere uma abordagem mais eficaz para o controle de tráfego. A velocidade média mais baixa do DQN é compensada por melhorias em outras métricas, resultando em um sistema de tráfego mais equilibrado e eficiente.

A Figura 6.6 mostra o gráfico da velocidade média versus o número de *steps* para os três sistemas.



FONTE: Elaboração própria

Figura 6.6 – *Velocidade Média vs. Step*

Os *scripts* para simulações no *SUMO*, a implementação dos algoritmos de aprendizado por reforço, as análises dos resultados obtidos, além de diagramas, gráficos e tabelas que auxiliam na compreensão das metodologias aplicadas, foram disponibilizados no *GitHub* (SILVA, 2024).

Capítulo 7

Sugestão para Implementação Real de Algoritmos de Aprendizagem por Reforço em Semáforos

A implementação de um sistema de controle de semáforos utilizando algoritmos de aprendizagem por reforço requer uma abordagem cuidadosamente planejada, levando em consideração aspectos técnicos e operacionais. A seguir estão os principais passos sugeridos para implementar essa tecnologia em um ambiente real:

7.1 Elementos Necessários

7.1.1 Infraestrutura de Comunicação

É crucial estabelecer uma infraestrutura de comunicação confiável entre os semáforos e uma central de controle. A recomendação é o uso de tecnologias como redes sem fio de alta capacidade (5G ou *Wi-Fi)* ou fibra óptica, garantindo baixa latência e alta taxa de transferência de dados. Em cenários de maior complexidade, considere a implementação de comunicação dedicada, como o sistema DSRC (*Dedicated Short Range Communication*), especialmente desenvolvido para aplicações de transporte. Essa infraestrutura permitirá que os semáforos troquem dados em tempo real sobre o estado do tráfego, facilitando uma resposta rápida e coordenada para otimizar o fluxo de veículos (AUTHOR1; AUTHOR2, 2023).

7.1.2 Sensores de Tráfego em Tempo Real

A base de um sistema de controle eficiente é a coleta precisa de dados de tráfego em tempo real. Para isso, recomenda-se a instalação de uma rede de sensores em pontos estratégicos das interseções. *Loops* indutivos no pavimento podem detectar a presença de veículos, enquanto câmeras e sistemas de *LIDAR* podem monitorar o fluxo de pedestres e fornecer dados mais detalhados sobre o ambiente. Esses dados serão utilizados pelo algoritmo de aprendizagem por reforço para identificar o estado atual do tráfego e tomar decisões sobre o tempo de sinalização. É importante que os sensores sejam confiáveis, de fácil manutenção, e que forneçam dados com alta precisão (AUTHOR1; AUTHOR2, 2023; AUTHOR5, 2023).

7.1.3 Controladores Locais para Processamento

Cada semáforo deverá ser equipado com um controlador local capaz de processar os dados fornecidos pelos sensores e aplicar as decisões tomadas pelo algoritmo de aprendizagem por reforço. Recomenda-se o uso de microprocessadores ou microcontroladores com capacidade de realizar cálculos em tempo real. Alternativamente, em áreas de maior complexidade ou densidade de tráfego, o processamento pode ser realizado de forma centralizada, onde um servidor de alto desempenho coleta os dados de vários semáforos e toma decisões coordenadas para otimizar o tráfego em uma área mais ampla (AUTHOR5, 2023).

7.1.4 Treinamento do Algoritmo em Simulações

Antes de implementar o sistema no mundo real, recomenda-se o treinamento extensivo do algoritmo de aprendizagem por reforço em um ambiente de simulação. O uso de ferramentas como o SUMO permitirá testar diferentes cenários de tráfego, como horas de pico, acidentes ou eventos especiais, sem riscos à segurança. Esse treinamento deve incluir uma variedade de condições de tráfego para que o algoritmo desenvolva políticas robustas e seja capaz de lidar com a variabilidade que encontrará no ambiente real (AUTHOR1; AUTHOR2, 2023).

7.1.5 Implementação de Processamento Centralizado ou Descentralizado

Dependendo da escala e da complexidade da área onde os semáforos serão implementados, deve-se decidir entre uma abordagem descentralizada ou centralizada para o processamento. Em áreas menores, o processamento descentralizado pode ser suficiente, com cada semáforo tomando suas próprias decisões com base nos dados locais. No entanto, em redes urbanas mais complexas, é recomendada a implementação de um sistema centralizado que permita uma coordenação mais eficiente entre vários semáforos, garantindo uma maior fluidez no trânsito em grandes áreas metropolitanas (AUTHOR3; AUTHOR4, 2019).

7.1.6 Mecanismos de Adaptação e Segurança

É fundamental que o sistema de semáforos inteligentes inclua mecanismos de segurança e adaptação. O algoritmo deve ser capaz de se adaptar a mudanças no fluxo de tráfego em tempo real, como bloqueios de vias ou a passagem de veículos de emergência. Para isso, recomenda-se a integração de módulos de detecção de anomalias que possam identificar situações atípicas e agir de forma segura, garantindo que o sistema mantenha a segurança dos pedestres e veículos em todas as circunstâncias (AUTHOR1; AUTHOR2, 2023; AUTHOR3; AUTHOR4, 2019).

A implementação real de algoritmos de aprendizagem por reforço em semáforos exige uma infraestrutura robusta e confiável, aliada a tecnologias avançadas de sensoriamento e processamento. O uso de simulações no treinamento do algoritmo é uma etapa crucial para garantir que o sistema funcione de forma eficiente em ambientes reais. Ao seguir essas recomendações, espera-se que o sistema de semáforos inteligentes seja capaz de melhorar significativamente o fluxo de tráfego nas cidades, reduzindo congestionamentos e otimizando os tempos de viagem.

Capítulo 8

Considerações Finais

Este trabalho teve como objetivo comparar a eficácia de semáforos inteligentes, controlados por algoritmos de aprendizado por reforço, com semáforos de tempo fixo tradicionais em redes de tráfego, conforme especificado no Manual Brasileiro de Sinalização de Trânsito – Volume V: Sinalização Semafórica. Utilizou-se o ambiente SUMO para simular o desempenho dos semáforos em dois cenários distintos: uma interseção simples e uma rede com três semáforos alinhados horizontalmente.

Os principais desafios encontrados durante a realização deste trabalho incluíram a complexidade dos algoritmos de aprendizado por reforço e a necessidade de maior poder computacional para o treinamento desses algoritmos. A configuração adequada dos parâmetros de aprendizado também se mostrou desafiadora, exigindo diversas experimentações para otimizar os resultados.

Os resultados obtidos indicaram que os semáforos inteligentes, especialmente aqueles utilizando o algoritmo *Deep Q-Network* (DQN), apresentaram melhorias significativas em vários parâmetros de desempenho. Em uma interseção simples, o sistema DQN apresentou a menor média de paradas totais, com 7.01 paradas, em comparação com 9.33 paradas para o sistema de tempo fixo. A redução no número de paradas implica diretamente em um fluxo de tráfego mais eficiente, reduzindo o tempo de viagem e melhorando a experiência dos motoristas. No cenário com três interseções alinhadas horizontalmente, observou-se que o DQN também proporcionou uma redução significativa no tempo total de espera dos veículos, mostrando-se mais eficiente em comparação com os semáforos de tempo fixo.

É importante destacar, contudo, que a análise da temporização fixa dos semáforos não se baseou em uma otimização manual exaustiva. A pesquisa focou principalmente em comparar os semáforos de aprendizado por reforço com semáforos de tempo fixo baseados em diretrizes padrão, e não em soluções ajustadas por especialistas em tráfego. Isso pode indicar um viés favorável à inteligência artificial, já que uma configuração otimizada manualmente poderia, potencialmente, apresentar resultados mais competitivos. Futuros estudos poderiam abordar essa lacuna, comparando os semáforos inteligentes com soluções manuais otimizadas por especialistas na área.

Essa abordagem adicional evitaria que o valor da tecnologia de inteligência artificial fosse superestimado meramente por sua inovação. Ao mesmo tempo, validaria de maneira mais robusta as conclusões sobre o desempenho superior dos semáforos inteligentes, assegurando que o uso da IA não é apenas uma escolha tecnológica, mas sim a solução mais eficaz para o problema.

Embora o algoritmo *Q-learning* tenha apresentado resultados melhores do que os semáforos de tempo fixo, ele requer um tempo de aprendizado consideravelmente maior. Foram necessários vários ajustes nos parâmetros e interrupções durante o processo de treinamento. Além disso, o *Q-learning* demanda muito mais treinamento comparado ao DQN, tornando impraticável sua utilização em tempo real devido à complexidade e ao tempo necessário para alcançar um desempenho otimizado.

Esses resultados confirmam a hipótese de que os métodos de aprendizado por reforço, especialmente o DQN, podem otimizar o controle de tráfego em ambientes urbanos complexos, proporcionando uma maior fluidez e reduzindo os congestionamentos. Além disso, os sistemas de controle de tráfego baseados em DQN mostraram-se promissores para uma operação de tráfego mais suave e sustentável, contribuindo para a diminuição do consumo de combustível e emissões de poluentes.

8.1 Sugestões para Trabalhos Futuros

Para futuras pesquisas, recomenda-se a investigação de outros algoritmos de aprendizado por reforço e a combinação desses métodos com técnicas de inteligência artificial, como redes neurais profundas e aprendizado por transferência. A implementação e avaliação em cenários reais, além de simulações, são essenciais para validar a eficácia das soluções propostas e adaptá-las às variáveis do mundo real.

Outra sugestão é a integração dos semáforos inteligentes com sistemas de transporte cooperativo e veículos autônomos, ampliando o potencial de otimização do fluxo de tráfego. Estudos adicionais poderiam também focar na análise de custo-benefício da implementação de tais sistemas, considerando não apenas os ganhos em eficiência, mas também os investimentos necessários em infraestrutura e tecnologia.

Adicionalmente, seria valioso conduzir experimentos que incluam a otimização manual dos semáforos de tempo fixo por especialistas em tráfego, como uma linha de base comparativa mais robusta. Isso ajudaria a garantir que as melhorias apresentadas pelas abordagens de inteligência artificial não sejam atribuídas apenas ao uso de uma tecnologia mais avançada, mas sim à sua eficácia genuína em relação a soluções tradicionais otimizadas.

Em resumo, este trabalho contribuiu para a compreensão dos benefícios e desafios associados ao uso de aprendizado por reforço em sistemas semafóricos, oferecendo uma base sólida para o desenvolvimento de soluções mais avançadas e eficientes no futuro.

REFERÊNCIAS

AUER, P.; CESA-BIANCHI, N.; FISCHER, P. Finite-time analysis of the multi-armed bandit problem. *Machine Learning*, v. 47, n. 2-3, p. 235–256, 2002. Acessado em: 15 abril 2024. Disponível em: <https://link.springer.com/article/10.1023/A:1013689704352>. (Citado na página 11.)

AUTHOR1, A.; AUTHOR2, B. Towards real-world deployment of reinforcement learning for traffic signal control. *Journal Name*, v. 12, n. 3, p. 123–130, 2023. Acessado em: 15 de Agosto de 2024. Disponível em: <https://link.springer.com/article/10.1007/s11036-021-01723-1>. (Citado 3 vezes nas páginas 59, 60, and 61.)

AUTHOR3, C.; AUTHOR4, D. Reinforcement learning for traffic light control with emphasis on emergency vehicles. *Journal of Supercomputing*, v. 28, p. 1451–1460, 2019. Acessado em: 15 de Agosto de 2024. Disponível em: <https://link.springer.com/article/10.1007/s11227-019-02988-x>. (Citado na página 61.)

AUTHOR5, E. Traffic light control with reinforcement learning. *Papers With Code*, 2023. Acessado em: 15 de Agosto de 2024. Disponível em: <https://paperswithcode.com>. (Citado na página 60.)

CHEN, H.; WANG, Y. et al. Exploiting semantic epsilon greedy exploration strategy in multi-agent reinforcement learning. *arXiv preprint arXiv:2201.10803*, 2022. Acessado em: 15 abril 2024. Disponível em: <https://arxiv.org/abs/2201.10803>. (Citado na página 11.)

CHEN, Y. et al. Traffic signal optimization control method based on adaptive weighted averaged double deep q network. *Applied Intelligence*, p. 1–22, 2023. Disponível em: <https://link.springer.com/article/10.1007/s10489-022-03377-5>. (Citado na página 18.)

DENATRAN. *Manual Brasileiro de Sinalização de Trânsito – Volume V: Sinalização Semafórica*. Departamento Nacional de Trânsito, 2020. Disponível em: <https://www.gov.br/transportes/pt-br/assuntos/transito/arquivos-senatran/docs/copy_of___05___MBST_Vol._V___Sinalizacao_Semaforica.pdf>. (Citado 5 vezes nas páginas 1, 17, 25, 27, and 31.)

Eclipse Foundation. *SUMO: Simulation of Urban MObility*. [S.l.], 2018. Acessado em: 20 abril 2024. Disponível em: <https://www.eclipse.org/sumo/>. (Citado na página 20.)

Eclipse Foundation. *TraCI: Traffic Control Interface*. [S.l.], 2024. Acessado em: 20 abril 2024. Disponível em: <https://sumo.dlr.de/docs/TraCI.html>. (Citado na página 21.)

Farama Foundation. *Gymnasium: A Toolkit for Developing and Comparing Reinforcement Learning Algorithms*. [S.l.], 2024. Acessado em: 25 abril 2024. Disponível em: <https://farama.org/Gymnasium/>. (Citado na página 22.)

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. MIT Press, 2016. Acesso em: 20 de abril de 2024. Disponível em: <http://www.deeplearningbook.org>. (Citado na página 16.)

GROSSI, E.; BUSCEMA, M. Introduction to artificial neural networks. *European journal of gastroenterology hepatology*, v. 19, p. 1046–54, 01 2008. Acessado 17 de abril 2024. (Citado na página 13.)

IN, B. Relu activation function explained. *Built In*, 2023. Acessado em: 10 de abril 2024. Disponível em: <https://builtin.com/machine-learning/relu-activation-function>. (Citado na página 14.)

Instituto de Pesquisa Econômica Aplicada. *Mobilidade Urbana no Brasil*. 2020. Acessado em: 20 abril 2024. Disponível em: <https://repositorio.ipea.gov.br/bitstream/11058/9186/1/Mobilidade%20urbana.pdf>. (Citado na página 1.)

KAELBLING, L. P.; LITTMAN, M. L.; MOORE, A. W. Reinforcement learning: A survey. *Journal of artificial intelligence research*, v. 4, p. 237–285, 1996. Disponível em: <https://archive.org/details/arxiv-cs9605103>. Acessado em: 15 abril 2024. (Citado 2 vezes nas páginas 5 and 8.)

KARTIKASARI, R.; PRAKARSA, G.; PRADEKA, D. Optimization of traffic light control using fuzzy logic sugeno method. *International Journal of Global Operations Research*, v. 1, n. 2, p. 51–61, 2020. Disponível em: <https://doi.org/10.23917/ijgor.v1i2.304>. (Citado na página 17.)

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. *Nature*, Nature Publishing Group, v. 521, n. 7553, p. 436–444, 2015. Acesso em: 20 de abril de 2024. Disponível em: <https://doi.org/10.1038/nature14539>. (Citado na página 12.)

LI, J. et al. Distributed signal control of arterial corridors using multi-agent deep reinforcement learning. *Transportation Research Part C: Emerging Technologies*, p. 146–164, 2023. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2352146522001624>. (Citado na página 18.)

LIN, L.-J. Self-improving reactive agents based on reinforcement learning, planning and teaching. In: ELSEVIER. *Machine Learning Proceedings 1992*. 1992. p. 292–305. Disponível em: <https://link.springer.com/article/10.1007/BF00992699>. (Citado na página 16.)

MELO, F. S. Convergence of q-learning: A simple proof. *Instituto de Engenharia de Sistemas e Computadores*, v. 1, 2001. Acesso em: 20 de abril de 2024. Disponível em: <https://www.inesc-id.pt/publications/286/pdf>. (Citado na página 12.)

MNIH, V. et al. Human-level control through deep reinforcement learning. *Nature*, Nature Publishing Group, v. 518, n. 7540, p. 529–533, 2015. (Citado 2 vezes nas páginas 12 and 16.)

MNIH, V. et al. Human-level control through deep reinforcement learning. *Nature*, Nature Publishing Group, v. 518, n. 7540, p. 529–533, 2015. Disponível em: <https://www.nature.com/articles/nature14236>. (Citado na página 41.)

MOHAMMADZADEH, A. et al. *Multilayer Perceptron (MLP) Neural Networks*. 2022. Acessado em: 15 de abril 2024. Disponível em: <https://doi.org/10.1007/978-3-031-14571-1_2>. (Citado na página 15.)

MURESAN, M.; FU, L.; PAN, G. Adaptive traffic signal control with deep reinforcement learning an exploratory investigation. *arXiv preprint arXiv:1901.00960*, 2019. Disponível em: <https://arxiv.org/abs/1901.00960>. (Citado na página 17.)

PUTERMAN, M. L. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 2014. Acessado em: 15 de abril 2024. Disponível em: <https://archive.org/details/markocvdecisionp0000unse>. (Citado na página 6.)

Raffin, Antonin and Hill, Ashley and Gleave, Adam and Kanervisto, Anssi and Ernestus, Maximilian and Dormann, Noah. *Stable Baselines3: Reliable Reinforcement Learning Implementations*. [S.l.], 2024. Acessado em: 25 abril 2024. Disponível em: <https://stable-baselines3.readthedocs.io/>. (Citado na página 22.)

SILVA, J. C. O. *Trabalho de Conclusão de Curso - Comparação de Semáforos Inteligentes com Aprendizado por Reforço e Semáforos de Tempo Fixo em Redes de Tráfego*. 2024. Acessado em: 10 de Agosto de 2024. Disponível em: <https://github.com/JefersonIFBA2017213024/TCC---Jeferson-Caio>. (Citado na página 58.)

SUTTON, R. S.; BARTO, A. G. *Reinforcement Learning: An Introduction*. MIT Press, 2018. Acessado em: 12 abril 2024. Disponível em: <https://web.stanford.edu/class/psych209/Readings/SuttonBartoIPRLBook2ndEd.pdf>. (Citado 4 vezes nas páginas 4, 5, 7, and 41.)

TAYLOR, M.; STONE, P. Adaptive traffic signal control using reinforcement learning. *AI Magazine*, Association for the Advancement of Artificial Intelligence, v. 28, n. 3, p. 27–38, 2016. (Citado na página 3.)

TOKIC, M. Adaptive ϵ-greedy exploration in reinforcement learning based on value differences. In: SPRINGER. *Annual conference on artificial intelligence*. 2010. p. 203–210. Acessado em: 15 de Agosto de 2024. Disponível em: <https://link.springer.com/chapter/10.1007/978-3-642-16111-7_29>. (Citado na página 41.)

WATKINS, C. J.; DAYAN, P. Q-learning. *Machine Learning*, v. 8, n. 3-4, p. 279–292, 1992. Acessado em: 12 de abril 2024. Disponível em: <https://link.springer.com/article/10.1007/BF00992698>. (Citado 5 vezes nas páginas 8, 9, 10, 11, and 41.)

ZHANG, K. et al. State representation learning for deep reinforcement learning. *arXiv preprint arXiv:1909.00125*, 2019. Acessado em: 12 abril 2024. Disponível em: <https://arxiv.org/pdf/1909.00125.pdf>. (Citado na página 5.)

ZHU, Z. et al. Transfer learning in deep reinforcement learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, p. 1–20, 2023. Disponível em: <https://doi.org/10.1109/TPAMI.2023.3216545>. (Citado na página 18.)